User Manual

# 3-Heights™ PDF to PDF/A Converter Shell

Version 4.5

PDF/A compliant

PDF-TOOLS.COM
Premium PDF Technology

# Contents


**1 Introduction . . . 1**
1.1 Description . . . 1
1.2 Functions . . . 2
1.3 Operating Systems . . . 3
1.4 Digital Signatures . . . 3
What is an Electronic Signature? . . . 4
How to Create Electronic Signatures . . . 5
**2 Installation . . . 7**
2.1 Windows . . . 7
2.2 Unix . . . 7
2.3 Note about the Evaluation Version . . . 8
**3 License Management . . . 8**
3.1 Graphical License Manager Tool . . . 8
3.2 Command Line License Manager Tool . . . 9
3.3 License Key Storage . . . 9
**4 User's Guide . . . 10**
4.1 Process Description . . . 10
4.2 What is PDF/A? . . . 12
4.3 Color Spaces . . . 13
4.4 Fonts . . . 15
4.5 Cryptographic Provider . . . 16
PKCS#11 Provider . . . 17
Microsoft CryptoAPI Provider . . . 18
3-Heights™ Signature Creation and Validation Service . . . 21
SwissSign Personal Signing Service . . . 23
SwissSign SuisseID Signing Service . . . 24
QuoVadis sealsign . . . 25
Swisscom All-in Signing Service . . . 26
4.6 How to Create Digital Signatures . . . 27
How to Create a PAdES LTV Signature . . . 27
How to Create a Visual Appearance of a Signature . . . 28
Miscellaneous . . . 28
4.7 How to Validate Digital Signatures . . . 30
**5 Reference Manual . . . 32**
5.1 General Settings . . . 32
-cl: Set Conformance . . . 32
-cem: Mask Conversion Errors . . . 33
-fd: Set font directory . . . 34
-id: Set value in the document information dictionary . . . 34
-ma: Analyze the Input File . . . 34
-mc: Force Conversion even if there Are Analysis Errors . . . 34
-mp: Post-Analyze the Result . . . 34
-p: Read an Encrypted PDF File . . . 35
-ax: Add XMP Metadata . . . 35
-ow: Optimize for the Web . . . 35
-q: Image Quality . . . 35
-rd: Report Conformance Violations in Detail . . . 35
-rs: Report Conformance Violations Summary . . . 36
-cff: Embed Type 1 fonts as CFF . . . 36
-uf: Update the fonts' Unicodes . . . 36
-ef: Add embedded file . . . 36
-af: Add associated file . . . 37
-az: Add a ZUGFeRD XML invoice file. . . . 37
-v: Verbose Mode . . . 37
-lk: Set License Key . . . 37
5.2 Processing Files in a Directory . . . 37
5.3 Color Profiles . . . 38
-cs: ICC Profile for Device-Specific Color Spaces . . . 38

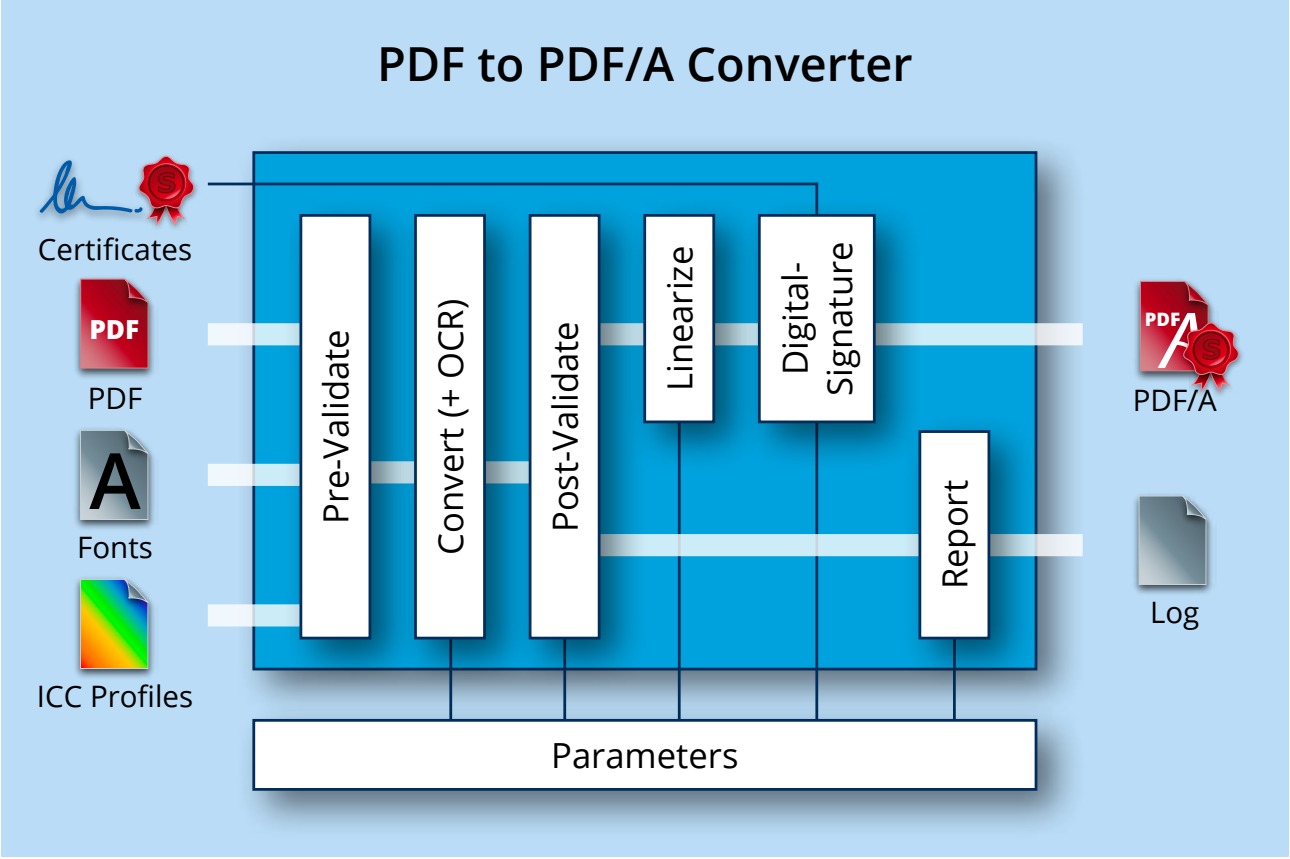
-oi: ICC Profile for Output Intent . . . 38
5.4 Digital Signatures . . . 38
-ap: Signature Page Number . . . 38
-ar: Signature Annotation Rectangle . . . 38
-cn: Certificate Name (Subject) . . . 38
-cr: Signature Reason . . . 39
-cci: Signer contact info . . . 39
-ca: Abort Conversion if Document Is Signed . . . 39
-ci: Certificate Issuer . . . 39
-cno: Certificate Serial Number . . . 39
-cfp: Certificate Fingerprint . . . 39
-co: Do not Embed Revocation Information . . . 40
-cp: Cryptographic Provider . . . 40
-cps: Cryptographic session property (string) . . . 40
-cpf: Cryptographic session property (file) . . . 40
-csl: Certificate Store Location . . . 41
-csn: Certificate Store Name . . . 41
-tsu: Time-stamp URL . . . 41
-tsc: Time-stamp Credentials . . . 41
-wpu: Web Proxy Server URL . . . 41
-wpc: Web Proxy Server Credentials . . . 41
-nc: Disable cache for CRL and OCSP . . . 41
-af1: Signature Font Name 1 . . . 41
-af2: Signature Font Name 2 . . . 42
-at1: Signature Text 1 . . . 42
-at2: Signature Text 2 . . . 42
-abg: Signature Background Image . . . 42
5.5 OCR . . . 42
-le: List OCR Engines . . . 42
-ocr: Load OCR Engine . . . 43
-ocl: Set OCR Language . . . 43
-ocp: Set OCR Parameters . . . 43
-ocs: Do Not Re-embed De-skewed Image . . . 43
-oci: Do not deskew image . . . 43
-ocd: Resolution for OCR Recognition . . . 43
-oct: Threshold Resolution for OCR . . . 44
-ocb: Convert Images to Bitonal before OCR Recognition . . . 44
-ocm OCR mode . . . 44
-oca: Rotate the image according to the detected angle . . . 44
-ocbc: Embed barcodes . . . 44
-ocx: Export recognized ocr text to file . . . 45
5.6 Return Codes . . . 46
**6 Log File . . . 47**
**7 Licensing and Copyright . . . 48**
**8 Contact . . . 48**

# 1 Introduction

## 1.1 Description

The 3-Heights™ PDF to PDF/A Converter Shell converts PDF files into PDF/A files. PDF/A has been acknowledged world-wide as the ISO standard for long-term archiving since 2005. The tool analyzes and converts the input file, applying a digital signature where required.

The integrated validator then optionally checks conformity once again. This product is robust and powerful and therefore predestined for archive migrations of any size.




## 1.2 Functions

The 3-Heights™ PDF to PDF/A Converter Shell accepts files from many different applications and automatically converts them into PDF/A. The level of conformity can be set to Level A or Level B. ICC color profiles for device-dependent color profiles and font types are embedded in the document. There is an option to provide the entire character set for fonts (no subsetting) to facilitate editing at a later stage. Missing fonts are reproduced as close to the original as possible via font recognition. Metadata can be generated automatically or added from external sources. The tool also detects and automatically repairs problems typical of the PDF format. A digital signature can be applied and a conformity check carried out at the end of the process. The optional OCR Add-On and linearization for fast web display are valuable additional functions.

## Features

- Conversion (PDF/A-1, PDF/A-2, PDF/A-3)
- Selectable level of conformity
- Embedding ICC color profiles for device-dependent color spaces
- Replace and subset fonts
- Validation
- File analysis and repair
- Conversion reporting
- Digital signatures, PDF/A-compliant
- Configure the virtual appearance of the signature (page, size, color, position, text, background image, etc.)
- Write the application log to a log file
- Enforce conversion even if the file is unconvertible
- Metadata management
- Read encrypted input files
- Encryption with access authorizations (not for PDF/A)
- Linearization (fast web display)
- JBIG2 compression
- JPEG2000 compression
- Conversion of embedded and attached files (PDF/A-2 and later)
- Colorants management (PDF/A-2 and later)
- OCR (optional)
- List OCR plug-Ins
- Set the OCR language
- Add embedded files (PDF/A-2) and associated files (PDF/A-3)
- Embedded XML invoice data conforming to the ZUGFeRD specification (PDF/A-3).

## Formats

Input Formats

- PDF 1.x (PDF 1.4, PDF 1.5, etc)

Target Formats:

- PDF/A-1a, PDF/A-1b
- PDF/A-2a, PDF/A-2b, PDF/A-2u
- PDF/A-3a, PDF/A-3b, PDF/A-3u

## Compliance

- Standards: ISO 19005-1 (PDF/A-1), ISO 19005-2 (PDF/A-2), ISO 19005-3 (PDF/A-3), ISO 32000 (PDF 1.7), PAdES Part 2
- Quality assurance: Isartor test suite

## 1.3 Operating Systems

- Windows XP, Vista, 7, 8, 8.1 - 32 and 64 bit
- Windows Server 2003, 2008, 2008 R2, 2012, 2012 R2 - 32 and 64 bit
- HP-UX 11 and later PA-RISC2.0 32 bit or HP-UX 11i and later ia64 (Itanium) 64 bit
- IBM AIX 5.1 and later (64 bit)
- Linux (32 and 64 bit)
- Mac OS X 10.4 and later (32 and 64 bit)
- Sun Solaris 2.8 and later, SPARC and Intel
- FreeBSD 4.7 and later 32 bit or FreeBSD 9.3 and later 64 bit (on request)

## 1.4 Digital Signatures

### Overview

Digital signature is a large and slightly complex topic. This manual gives an introduction to digital signatures and describes how the 3-Heights™ PDF to PDF/A Converter Shell is used to apply them. It does however not describe all the technical details.

### Terminology

*Digital Signature* is a cryptographic technique of calculating a number (a digital signature) for a message. Creating a digital signature requires a private key from a certificate. Validating a digital signature and its authorship requires a public key. Digital Signature is a technical term.

*Electronic Signature* is a set of electronic data that is merged or linked to other electronic data in order to authenticate it. Electronic Signatures can be created by means of a digital signature or other techniques. Electronic Signature is a legal term.

| Table: Abbreviations | |
|---|---|
| CA | Certification Authority |
| CMS | Cryptographic Message Syntax |
| CRL | Certificate Revocation List |
| CSP | Cryptographic Service Provider |
| HSM | Hardware Security Module |
| OCSP | Online Certificate Status Protocol |
| PKCS | Public Key Cryptography Standards |
| QES | Qualified Electronic Signature |
| TSA | Time-stamp Authority |
| TSP | Time-stamp Protocol |

## Why Digitally Signing?

The idea of applying a digital signature in PDF is very similar to a handwritten signature: A person reads a document and signs it with their name. In addition to the name, the signature can contain further optional information, such as the date and location. A valid electronic signature is a section of data that can be used to:

- Ensure the integrity of the document
- Authenticate the signer of the document
- Prove existence of file prior to date (time-stamp)

Digitally signing a document requires a certificate and its private key. How to access and use a certificate is described in the chapter Cryptographic Provider.

In a PDF document, a digital signature consists of two parts:

- *A PDF related part*: This part consists of the PDF objects required to embed the signature into the PDF document. This part depends on the signature type (Document Signature, MDP Signature, see table below). Information such as name of the signer, reason, date, location is stored here. The signature may optionally have a visual appearance on a page of the PDF document, which can contain text, graphics and images. This part of the signature is entirely created by the 3-Heights™ PDF to PDF/A Converter Shell.
- *A cryptographic part*: A digital signature is based on a cryptographic checksum (hash value) calculated from the content of the document that is being signed. If the document is modified at a later time, the computed hash value is no longer correct and the signature becomes invalid, i.e. the validation will fail and will report that the document has been modified since the signature was applied. Only the owner of the certificate and its private key is able to sign the document. However, anybody can verify the signature with the public key contained in the certificate. This part of the signature requires a cryptographic provider for some cryptographic data and algorithms.

The 3-Heights™ PDF to PDF/A Converter Shell supports the following types of digital signatures:

- *Document Signature*: Check the integrity of the signed part of the document and authenticate the signer's identity. One or more signatures can be applied. A signed document can be modified and saved by incremental update. The state of the document can be re-created as it existed at the time of signing.
- *MDP (Modification detection and prevention) Signature*: Enable detection of disallowed changes specified by the author. A document can contain only one MDP signature; it must be the first in the document. Other document signatures may be present.
- *Document Time-stamp Signature*: Establish the exact content of the file at the time indicated by the Time-stamp. One or more document Time-stamp signatures can be applied. A signed document can be modified and saved by incremental update.

### 1.4.1 What is an Electronic Signature?

There are different types of electronic signatures, which normally are defined by national laws, and therefore are different for different countries. The type of electronic signatures required in a certain process is usually defined by national laws. Quite advanced in this manner are German-speaking countries where such laws and an established terminology exist. The English terminology is basically a translation from German. Three types of electronic signatures are distinguished:

- Simple Electronic Signature "Einfache Elektronische Signatur"
- Advanced Electronic Signature "Fortgeschrittene Elektronische Signatur"
- Qualified Electronic Signature (QES) "Qualifizierte Elektronische Signatur"

All applied digital signatures are PDF/A and PAdES compliant.

#### Simple Electronic Signature

A simple electronic signature requires any certificate that can be used for digital signing. The easiest way to retrieve a certificate, which meets that requirement, is to create a so called self-signed certificate. Self-signed means it is signed by its owner, therefore the issuer of the certificate and the approver of the legitimacy of a document signed by this certificate is the same person.

*Example:* Anyone could create a self-signed certificate issued by "Peter Pan" and issued to "Peter Pan". Using this certificate one is able to sign in the name of "Peter Pan".

If a PDF document is signed with a simple electronic signature and the document is changed after the signature has been applied, the signature becomes invalid. However, the person who applied the changes, could at the same time (maliciously) also remove the existing simple electronic signature and - after the changes - apply a new, equally looking Simple Electronic Signature and falsify its date. As we can see, a simple electronic signature is neither strong enough to ensure the integrity of the document nor to authenticate the signer.

This drawback can be overcome using an Advanced or Qualified Electronic Signature.

#### Advanced Electronic Signature

Requirements for advanced certificates and signatures vary depending on the country where they are issued and used.

An advanced electronic signature is based on an advanced certificate that is issued by a recognized certificate authority (CA) in this country, such VeriSign, SwissSign, QuoVadis. In order to receive an advanced certificate, its owner must prove his identity, e.g. by physically visiting the CA and presenting his passport. The owner can be an individual, a legal person or another entity.

An advanced certificate contains the name of the owner, the name of the CA, its period of validity and other information.

The private key of the certificate is protected by a PIN, which is only known to its owner.

This brings the following advantages over a simple electronic signature:

- The signature authenticates the signer.
- The signature ensures the integrity of the signed content.

#### Qualified Electronic Signature

Requirements for qualified certificates and signatures vary depending on the country where they are issued and used.

A Qualified Electronic Signature is similar to an advanced electronic signature, but has higher requirements. The main differences are:

- It is based on a qualified certificate, which is provided as a hardware token (USB stick, smart card).
- For every signature it is required to enter the PIN code manually. This means that only one signature can be applied at a time.
- Certificate revocation information (OCSP/CRL) can be acquired from an online service. The response (valid, revoked, etc.) must be embedded in the signature.
- A time-stamp (TSP) that is acquired from a trusted time server (TSA) may be required.

This brings the following advantages over an advanced electronic signature:

- The signature ensures the certificate was valid at the time when the document was signed (due to the embedding of the OCSP/CRL response).
- The signature ensures the integrity of the time of signing (due to the embedding of the time-stamp).
- Legal processes that require a QES are supported.

Note that a Time-stamp can be added to any type of signature. OCSP/CRL responses are also available for some advanced certificates.

### 1.4.2 How to Create Electronic Signatures

#### Preparation Steps

1. Identify whether an advanced or a qualified signature is required. For most automated processes an advanced signature is sufficient.
2. Acquire a corresponding certificate from a CA. Note that some CA offer USB sticks or smart cards that contain both, an advanced and a qualified certificate.
3. Setup and configure the certificate's Cryptographic Provider
   - In case the certificate resides on hardware such as an USB token or a Smart Card, the required middleware (driver) needs to be installed.

- In case the certificate is a soft certificate, it must be imported into the certificate store of a cryptographic provider.

4. Optional: Acquire access to a trusted time server (TSA) (e.g. from the CA of your signing certificate).
5. Apply the signature by providing the following information:
   - Values for the selection of the signing certificate (e.g. the name of the certificate)
   - The Cryptographic Provider where the certificate is located
   - Optional: Time-stamp service URL (e.g. "http://server.mydomain.com:80/tsa")
   - Optional: Time-stamp service credentials (e.g. username:password)
   - Optional: Embed revocation information (default: true)
   - Optional: Visual appearance of the signature on a page of the document (e.g. an image).

### Example: Steps to Add an Electronic Signature

The 3-Heights™ PDF to PDF/A Converter Shell applies PDF/A compliant signatures. This means if a PDF/A document is digitally signed, it remains PDF/A compliant.

In order to add an electronic signature with the 3-Heights™ PDF to PDF/A Converter Shell the following steps need to be done:

1. Provide the certificate name (Subject)
2. Apply settings for the signature, such as the reason text, or the visual appearance (color, position, etc).
3. Process the PDF document by a user which has access to the selected certificate and thereby add the signature

The certificate name is provided with the switch -cn, the reason with the switch -cr and the provider (including the PIN to access the certificate's private key) with the switch -cp. A sample command looks like this:

```
pdf2pdf
-cn "Philip Renggli"
-cp "cvp11.dll;0;secret-pin"
-cr "I reviewed the document"
-tsu "http://server.mydomain.com:80/tsa"
input.pdf output.pdf
```

Note: On Non-Windows platforms additional settings are required to be set, such as fonts.

The visual appearance of the digital signature on a page of the resulting output-document looks as shown below:

# 2 Installation

## 2.1 Windows

The retail version of the 3-Heights™ PDF to PDF/A Converter Shell comes as a ZIP archive containing various files including runtime binary executable code, files required for the developer, documentation and license terms.

1. Download the ZIP archive of the product from your download account at http://www.pdf-tools.com.
2. Unzip the file using a tool like WinZip available from WinZip Computing, Inc. at http://www.winzip.com to a directory on your hard disk where your program files reside (e.g. *C:\Program Files\PDF Tools AG*).
3. Check the appropriate option to preserve file paths (folder names). The unzip process now creates the following subdirectories:

   *bin:* Contains the runtime executable binary code.

   *bin\fonts:* Contains the PDF standard fonts and the font mapping file. Copy and thereby install the fonts to the OS fonts directory (*%systemroot%\fonts*, e.g. *C:\Windows\fonts*).

   *bin\icc:* Contains the two color profiles "USWebCoatedSWOP.icc" and "sRGB Color Space Profile.icm".

   *doc:* Contains documentation files.

   There is the option to download the software as MSI file, which makes the installation easier.
4. To easily use the 3-Heights™ PDF to PDF/A Converter Shell from a shell, the directory needs to be included in the "Path" environment variable.
5. Optionally register your license key using the License Manager.
6. Make sure your platform meets the requirements regarding color spaces and fonts described in chapters Color Profiles and Fonts respectively.
7. If you want to sign documents, proceed with setting up your cryptographic provider as described in chapter Cryptographic Provider.

## 2.2 Unix

This section describes installation steps required on all Unix platforms, which includes Linux, Mac OS X, Sun Solaris, IBM AIX, HP-UX, FreeBSD and others.

### All Unix Platforms

1. Unpack the archive in an installation directory, e.g. */opt/pdf-tools.com/*
2. Copy or link the executable into one of the standard executable directories, e.g:
   `ln -s /opt/pdf-tools.com/bin/pdf2pdf /usr/bin`
3. Verify that the GNU shared libraries required by the product are available on your system:
   - *On Linux*: `ldd pdf2pdf`
   - *On AIX*: `dump -H pdf2pdf`

   In case you have not installed the GNU shared libraries yet, proceed as follows:
   (a) Go to http://www.pdf-tools.com and navigate to "Support" → "Resouces".
   (b) Download the GNU shared libraries for your platform.
   (c) Extract the archive and copy or link the libraries into your library directory, e.g */usr/lib* or */usr/lib64*.
   (d) Verify that the GNU shared libraries required by the product are available on your system now.
4. Optionally register your license key using the Command Line License Manager Tool.
5. Make sure your platform meets the requirements regarding color spaces and fonts described in chapters Color Profiles and Fonts respectively.
6. If you want to sign documents, proceed with setting up your cryptographic provider as described in chapter Cryptographic Provider.

### How to set the Environment Variable "Path"

To set the environment variable "Path" on Windows, go to Start ->Control Panel (classic view) ->System ->Advanced ->Environment Variables.

Select "Path" and Edit, then add the directory where *pdf2pdf.exe* is located to the "Path". If the environment variable "Path" does not exist, create it.

## 2.3 Note about the Evaluation Version

The evaluation versions of the 3-Heights™ products automatically add a watermark to the output files.

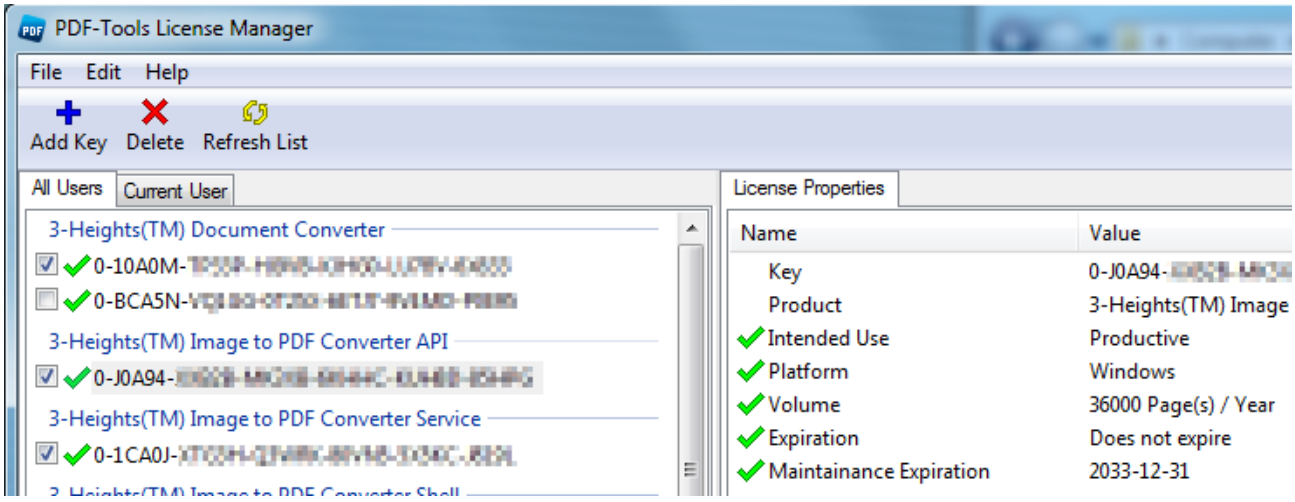
# 3 License Management

There are three possibilities to pass the license key to the application:

1. The license key is installed using the GUI tool (Graphical user interface). This is the easiest way if the licenses are managed manually. It is only available on Windows.
2. The license key is installed using the shell tool. This is the preferred solution for all non-Windows systems and for automated license management.
3. The license key is passed to the application at runtime via the switch `-lk`. This is the preferred solution for OEM scenarios.

## 3.1 Graphical License Manager Tool

The GUI tool *LicenseManager.exe* is located in the *bin* directory of the product kit.




### List all installed license keys

The license manager always shows a list of all installed license keys in the left pane of the window. This includes licenses of other PDF Tools products. The user can choose between:

- Licenses available for all users. Administrator rights are needed for modifications.
- Licenses available for the current user only.

### Add and delete license keys

License keys can be added or deleted with the "Add Key" and "Delete" buttons in the toolbar.

- The "Add key" button installs the license key into the currently selected list.
- The "Delete" button deletes the currently selected license keys.

### Display the properties of a license

If a license is selected in the license list, its properties are displayed in the right pane of the window.

### Select between different license keys for a single product

More than one license key can be installed for a specific product. The checkbox on the left side in the license list marks the currently active license key.

## 3.2 Command Line License Manager Tool

The command line license manager tool `licmgr` is available in the bin directory for all platforms except Windows.

A complete description of all commands and options can be obtained by running the program without parameters:

```
licmgr
```

### List all installed license keys

```
licmgr list
```

### Add and delete license keys

Install new license key:

```
licmgr store X-XXXXX-XXXXX-XXXXX-XXXXX-XXXXX-XXXXX
```

Delete old license key:

```
licmgr delete X-XXXXX-XXXXX-XXXXX-XXXXX-XXXXX-XXXXX
```

Both commands have the optional argument `-s` that defines the scope of the action:

- `g`: For all users
- `u`: Current user

### Select between different license keys for a single product

```
licmgr select X-XXXXX-XXXXX-XXXXX-XXXXX-XXXXX-XXXXX
```

## 3.3 License Key Storage

Depending on the platform the license management system uses different stores for the license keys.

### Windows

The license keys are stored in the registry:

- `HKLM\Software\PDF Tools AG` (for all users)
- `HKCU\Software\PDF Tools AG` (for the current user)

### Mac OS X

The license keys are stored in the file system:

- `/Library/Application Support/PDF Tools AG` (for all users)
- `~/Library/Application Support/PDF Tools AG` (for the current user)

### Unix/Linux

The license keys are stored in the file system:

- `/etc/opt/pdf-tools` (for all users)
- `~/.pdf-tools` (for the current user)

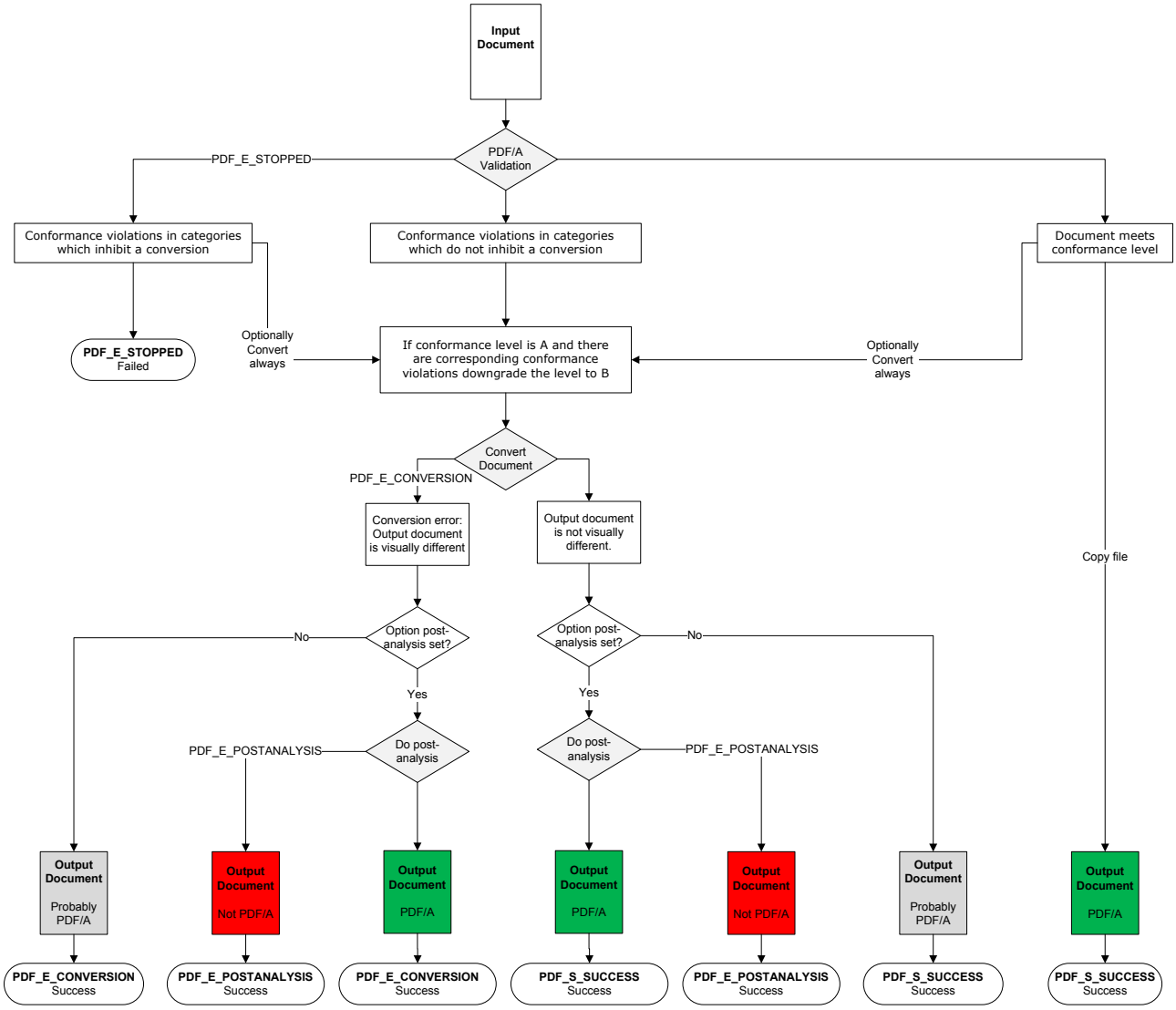
Note: The user, group and permissions of those directories are set explicitly by the license manager tool. It may be necessary to change permissions to make the licenses readable for all users. Example:

```
chmod -R go+rx /etc/opt/pdf-tools
```

# 4 User's Guide

## 4.1 Process Description

The workflow of the PDF to PDF/A Conversion is outlined in the graphic below.




1. *License Check*: The license is checked.
2. *Pre-Analysis*: The input document is analyzed. If the document is already compliant to the requested standard it is copied.
   If the required standard cannot be met, but a lower level of the standard can, the target level is downgraded, e.g. from PDF/A-1a to PDF/A-1b[1].
   It the target standard is PDF/A-1 and a file contains transparency or other elements that cannot be converted to PDF/A-1, the target standard is upgraded to PDF/A-2[2].
   If the input document contains non-convertible elements the conversion is stopped, except if convert-always is enabled.
3. *Conversion*: The actual conversion is performed.
   A failure indicates a case where no output file is created. Possible causes of that are: the output file cannot be written, or conversion was aborted.
   Success indicates a correct execution of the function and creation of an output document. However it does not imply there were no problems during the process. If actions had to be taken that might have altered

[1] Automatic downgrades can be deactivated using the option -cem.
[2] Automatic upgrades can be deactivated using the option -cem.

the visual appearance of the file or crucial data had to be removed, a conversion error is generated (see chapter “Conversion Errors” below).

4. *Post-Analysis*: If post analysis is enabled, the resulting PDF document is validated. If the resulting document does not meet the requested standard a post-analysis error is raised.

The above graphic includes PDF/A related errors only. Errors with digital signatures are not reflected in this workflow chart.

## Conversion Steps

The goal of the conversion is to create a document which is conforming to the PDF/A ISO standard.

If the analysis of the document indicates a conversion to the requested standard is possible, the following steps are performed:

- Embed and subset non-embedded font programs
- Replace device specific color spaces with CIE-based color spaces
- Add a GTS_PDFA output intent
- Remove prohibited entries
- Remove entries with a default value
- Remove entries with unknown values
- Add mandatory entries
- Add XMP metadata if missing or fix inconsitent XMP metadata
- Apply implicit optimization functions (e.g. replace and subset embedded fonts)
- Apply implicit repair functions (to conform with ISO19005-1 chapter 6.1)

If the analysis indicates a conversion is not possible, a “best effort” conversion can be forced. In this case the output may or may not be PDF/A conformant. Use the post analysis feature in order to detect, whether or not the output is conformant. It is also possible that the output file looks visually different to the input file due to the forced conversion.

## Conversion Errors

The conversion error (return code 5) indicates that during conversion actions had to be taken that might have altered the visual appearance of the file or crucial data had to be removed. Use the option -cem to define what is crucial to your process and should therefore lead to a conversion error. Note that the resulting document is PDF/A compliant nonetheless.

The following issues may result in a conversion error:

- Optional content removed
- FFilter or FDecodeParms removed
- Prohibited annotation type converted to text annotation
- Prohibited action removed
- Embedded files removed
- Annotation without appearance stream
- Transparency removed
- Character from show string removed because glyph missing in font
- Unconvertible metadata

For a complete list of conversion events that can lead to a conversion error see option -cem.

Some of these conversion errors, such as transparency or optional content may be resolved by creating PDF/A-2 or PDF/A-3 instead of PDF/A-1.

### Post Analysis

Optionally a post analysis step can be activated in order to check, whether or not the output file created conforms to the requested standard. A post analysis error (return code 6) indicates that the output file is not PDF/A. In case of a post analysis error you can repeat the conversion with the option -rd and look at the log file in order to see why the post analysis failed.

## 4.2 What is PDF/A?

### PDF/A-1

The PDF/A-1 format is described in the international standard ISO-19005-1. It bases on the PDF 1.4 reference and has some additional requirements. Best is to have a general understanding of PDF/A. Here is a brief overview of how to create a PDF/A document from a non-PDF/A document.

1. A PDF/A has requirements about meta data and the structure of the file. The PDF to PDF/A Converter takes care of this and the user does not have to apply any settings. However he can provide the XMP meta data himself if desired.
2. In PDF/A, colors (including grayscale and black/white) must not be represented in a device color space (DeviceRGB, DeviceCMYK, DeviceGray). Suitable default color space profiles to substitute the device color spaces, one for RGB, CMYK and grayscale respectively can be provided by the user. In addition, or alternatively, one color space profile can be embedded as output intent. In this latter case, device colors are automatically managed by the output intent if the color can be represented in the space given by the color space profile in the output intent.
   If the converter encounters unmanaged colors, e.g. because no color space profile was set, then a calibrated color space is generated automatically, one RGB and one grayscale, for RGB and grayscale colors respectively. If unmanaged CMYK colors are encountered, a default CMYK output intent is embedded.
3. Fonts used in visible text must be embedded. This is automatically done by the Converter.
4. For PDF/A 1a: The original document structure information will be retained when converting the file to PDF/A. However, new tags will not be added and the structure will not be changed. To create a PDF/A-1a compliant file, the original file must have been created with the required structure and tagging. Otherwise, a PDF/A-1b file will be produced.

### What is the difference between PDF/A-1b and PDF/A-1a?

PDF/A-1a has additional specifications on top of PDF/A-1b. These are:

1. The encoding of fonts must meet additional requirements, e.g. include a ToUnicode mapping (ISO 19005-1, chapter 6.3.8)
2. The document must contain a logical structure (ISO 19005-1, chapter 6.8)

The idea of the PDF/A-1a requirements is mainly to provide support for disabled people, i.e. by providing the required information needed for applications that support the read out loud feature.

The logical structure of the document is a description of the content of the pages. This description has to be provided by the creator of the document. It consists of a fine granular hierarchical tagging that distinguishes between the actual content and artifacts (such as page numbers, footers, layout artifacts, etc.). The tagging provides a meaningful description. Examples are “This is a Header”, “This color image shows a small sailing boat at sunset”, etc. One can easily understand this information cannot be generated automatically, it needs to be provided. This is one of the reasons why not every PDF document can be converted to PDF/A-1a.

### PDF/A-2

PDF/A-2 is described in ISO 19005-2. It is based on ISO 32000-1, the standard for PDF 1.7. PDF/A-2 is meant as an extension to PDF/A-1. The second part shall complement the first part and not replace it. The most important differences between PDF/A-1 and PDF/A-2 are:

- The list of compression types has been extended by JPEG2000
- Transparent contents produced by graphic programs are allowed
- Optional contents (also known as layers) can be made visible or invisible
- Multiple PDF/A files can be bundled in one file (collection, package)
- The additional conformity level U (Unicode) allows for creating searchable files without having to fulfill the strict requirements of the conformity level A (accessibility)

Documents that contain features described above, in particular layers or transparency, should therefore be converted to PDF/A-2 rather than PDF/A-1.

### PDF/A-3

PDF/A-3 is described in ISO 19005-3. It is based on ISO 32000-1, the standard for PDF 1.7. PDF/A-3 is an extension to PDF/A-2. The third part shall complement the second part and not replace it. The only two differences between PDF/A-2 and PDF/A-3 are:

- Files of any format and conformance may be embedded. Embedded files need not be suitable for long-term archiving.
- Embed files can be associated with any part of the PDF/A-3 file.

For additional information about PDF/A please visit: http://www.pdf-tools.com/pdf/pdfa-longterm-archiving-iso-19005-pdf.aspx.

## 4.3 Color Spaces

### Colors in PDF

The PDF format supports a range of color spaces:

- *Device Color Spaces: DeviceGray, DeviceRGB, DeviceCMYK.* These are also referred to as uncalibrated color spaces, because they cannot be used to specify color values such that colors are reproducible in a predictable way on multiple output devices.
- *CIE-based Color Spaces: CalGray, CalRGB, Lab, ICCBased.* These are also referred to as device-independent color spaces, because they are inherently capable of specifying colors which can be reliably reproduced on multiple output devices.
- *Special Color Spaces: Separation and DeviceN.* These require an alternate color space from one of the previous two groups to allow the PDF consumer to simulate the color on devices which do not support the special color space.

Colors can occur in the following objects of a PDF/A document:

- Raster images (also inline images)
- Text and Vector objects such as lines and curves
- Annotations
- Shading patterns
- Transparency blending (PDF/A-2 and later)

## ICC Color Profiles

An ICC (International Color Consortium) profile is a file format which can be used to describe the color characteristics of a particular device. For example for the correct color reproduction when an image from a scanner or camera is displayed on a device, such as a monitor or printer. Color profiles are usually provided with the operating system (OS), on a Windows System, they can be found at the following location:
*%SystemRoot%\system32\spool\drivers\color*

Alternatively, profiles can be found here:

- www.pdf-tools.com/public/downloads/resources/colorprofiles.zip
- www.color.org/srgbprofiles.html
- www.adobe.com/support/downloads/iccprofiles/icc_eula_win_dist.html

Please note that most color profiles are copyrighted, therefore you should read the license agreements on the above links before using the color profiles. The PDF to PDF/A Converter will try to locate color profiles automatically in the %SystemRoot%\system32\spool\drivers\color folder as needed. On Unix platforms, you can store the color profiles contained in the "colorprofiles.zip" download in a folder of your choice, and set the environment variable PDF_ICC_PATH to point to that folder.

## PDF/A Requirements

In PDF/A the usage of uncalibrated color spaces (DeviceGray, DeviceRGB, and DeviceCMYK) is prohibited because colors that are specified in this way cannot be reproduced reliably on multiple output devices. Therefore, when converting to PDF/A, all device color spaces should be replaced by CIE-based color spaces. There is one exception to this rule: An uncalibrated color is tolerated if the output intent holds an ICC color profile with which this color can be represented. (E.g. a grayscale color can be represented in an RGB color profile, but a CMYK color cannot.)

The 3-Heights™ PDF to PDF/A Converter Shell uses the following strategy:

- For each device color space (DeviceGray, DeviceRGB, and DeviceCMYK) an ICC color profile can be specified to be used as substitute for the respective device color space.
- One ICC color profile can be set to be used in the output intent.
- During conversion, if a device color space is encountered then the following is done:
  - If an output intent was set that is capable of managing this color, no action is needed.
  - Otherwise, if an ICC color profile is set to substitute this device color space then this color profile is used.
  - Otherwise, for DeviceRGB and DeviceGray color spaces: A calibrated color space (CalRGB[3] and CalGray respectively) is generated and used as a substitute.
  - Otherwise, for DeviceCMYK color spaces:
    - If the output intent is not set, then a default CMYK ICC color profile is used for the output intent.
    - If the output intent holds a non-CMYK ICC color profile, then a default CMYK ICC color profile is generated and used as a substitute for DeviceCMYK.

The above strategy is motivated by the fact that CalRGB and CalGray color spaces occupy very little memory in comparison to ICC color profiles. Also note that the primary purpose of the output intent in a PDF document is to describe the characteristics of the device on which a document is intended to be rendered. Traditionally, the target device is a printer, which motivates CMYK output intents. The default CMYK color profile *USWebCoatedSWOP.icc* is provided in the sub-directory *bin\icc*.

[3]The generated CalRGB color space is an approximation to the ICC color profile *sRGB Color Space Profile.icm*.

## 4.4 Fonts

The PDF/A standard requires all fonts to be embedded in the PDF file. This ensures that the future rendering of the textual content of a conforming file matches, on a glyph by glyph basis, the appearance of the file as originally created.

Hence, if non-embedded fonts in a PDF are used, the font must be embedded. For this, a matching font has to be found in the font directories. The the option `-fd` should be used to define your font directories. The default font directories are listed in the documentation to the the option `-fd`.

It is important that the font directories contain all fonts that are used for the input files. In particular, the font ZapfDingbats is widely used in PDF documents, but not available on most systems. Therefore, the product kit includes this font, which should be added to a font directory.

Fonts should be added to one of the font directories, if the post analysis returns validation errors like the following: `"output.pdf", 9, 20, 0x00418704, "The font ShinGo must be embedded.", 1`

The fonts of the font directories and their properties are cached in a font cache, located in the files `font-database*` in the temporary files folder. The cache files have to be removed manually, if fonts are added or removed from these directories.

The directory for temporary files is determined as follows. The product checks for the existence of environment variables in the following order and uses the first path found:

- *Windows:*
  1. The path specified by the *TMP* environment variable.
  2. The path specified by the *TEMP* environment variable.
  3. The path specified by the *USERPROFILE* environment variable.
  4. The Windows directory.
- *Unix:*
  1. The path specified by the *PDFTMPDIR* environment variable.
  2. The path specified by the *TMP* environment variable.
  3. The */tmp* directory.

**Example:** Command to remove the font cache files on Windows

```
del %TMP%\font-database*
```

### Microsoft Core Fonts on Unix

Many PDF documents use Microsoft core fonts like Arial, Times New Roman and other fonts commonly used on Windows. Therefore, it is recommended to install these fonts to your default font directories. Many Linux distributions offer an installable package for these “Microsoft TrueType core fonts”. For instance, on Debian based systems the package is called `ttf-mscorefonts-installer`.

Alternatively you can download the fonts from here:
http://corefonts.sourceforge.net/

Microsoft has an FAQ on the subject, that covers licensing related questions as well:
http://www.microsoft.com/typography/faq/faq8.htm

### Font Configuration File fonts.ini

The font configuration file can be used to control the embedding of fonts. The file fonts.ini must reside at the following location, which is platform dependent:

- *Windows:* In a directory named “Fonts”, which must be a direct sub-directory of where the main DLL or executable resides.
- *Unix:* The “fonts.ini” file is searched in the following locations
  1. If the environment variable PDFFONTDIR is defined: `$PDFFONTDIR/fonts.ini`
  2. `~/.pdf-tools/fonts/fonts.ini`
  3. `/etc/opt/pdf-tools/fonts/fonts.ini`

fonts.ini uses the ini file format and has two sections. The section "fonts" is ignored by the PDF to PDF/A Converter, so you may remove it. In the section "replace" font replacement rules of the form key=value can be defined. The key specifies the font that is to be replaced. The key should match the name of the font mentioned in the pre-analysis of the PDF to PDF/A Converter. e.g. "ShingGo" for:

```
"file.pdf", 9, 20, 0x00418704, "The font ShinGo must be embedded.", 1
```

The value should match the true type name of an installed font. Do not replace any standard fonts (Helvetica, Arial, Times, TimesNewRoman, Courier, CourierNew, Symbol, and ZapfDingbats).

Please note that this feature should be used with care. Replacing a font with another might change the visual appearance of the file because of different glyph shapes or glyphs that are not available in the replacement font. Embedding another font might also have legal consequences.

**Example:** Replace MS-Mincyo with MS-Mincho

```
[replace]
MS-Mincyo=MS-Mincho
```

This rule defines, that in order to embed a font program for font MS-Mincyo the font MS-Mincho should be used. This rule is useful, because both names are possible transliterations of the same Japanese font. However, the official transliteration used by the actual font is MS-Mincho.

## 4.5 Cryptographic Provider

In order to use the 3-Heights™ PDF to PDF/A Converter Shell's cryptographic functions such as creating digital signatures, a cryptographic provider is required. The cryptographic provider manages certificates, their private keys and implements cryptographic algorithms.

The 3-Heights™ PDF to PDF/A Converter Shell can use various different cryptographic providers. The following list shows, for which type of signing certificate which provider can be used.

**USB Token or Smart Card**

These devices typically offer a PKCS#11 interface, which is the recommended way to use the certificate → PKCS#11 Provider.

On Windows, the certificate is usually also available in the Microsoft CryptoAPI. This provider is not recommended, unless you experience problems with your device's PKCS#11 interface.

If you need to sign documents on a non-Windows system with an USB token that does not come with middleware for your platform, you can use the 3-Heights™ Signature Creation and Validation Service.

If you need to sign documents on Windows in a non-interactive or locked session[4], use the 3-Heights™ Signature Creation and Validation Service.

**Hardware Security Module (HSM)**

HSMs always offer very good PKCS#11 support → PKCS#11 Provider

For more information and installation instructions see separate document TechNotePKCS11.pdf.

**Soft Certificate**

Soft certificates are typically PKCS#12 files that have the extension `.pfx` or `.p12` and contain the signing certificate as well as the private key and trust chain (issuer certificates). Soft certificate files cannot be used directly. Instead, they must be imported into the certificate store of a cryptographic provider.

- *All Platforms*: The recommended way of using soft certificates is to import them into a store that offers a PKCS#11 interface and use the PKCS#11 Provider. For example:
  - A HSM
  - openCryptoki on Linux
  - PKCS#11 softtoken on Solaris

  For more information and installation instructions of the above stores see separate document TechNotePKCS11.pdf.
- *Windows*: If no PKCS#11 provider is available, soft certificates can be imported into Windows certificate store, which can then be used as cryptographic provider → Microsoft CryptoAPI

[4] See the description of the 3-Heights™ Signature Creation and Validation Service for more details on this topic.

**Signature Service**

Signature services are a convenient alternative to storing certificates and key material locally. The 3-Heights™ PDF to PDF/A Converter Shell can use various different services whose configuration is explained in the following sections of this documentation:

- 3-Heights™ Signature Creation and Validation Service
- SwissSign Personal Signing Service
- SwissSign SuisseID Signing Service
- QuoVadis sealsign
- Swisscom All-in Signing Service

### 4.5.1 PKCS#11 Provider

PKCS#11 is a standard interface offered by most cryptographic devices such as HSMs, USB Tokens or sometimes even soft stores (e.g. openCryptoki).

More information on and installation instructions of the PKCS#11 provider of various cryptographic devices can be found in the separate document TechNotePKCS11.pdf.

#### Configuration

**Provider** Option -cp

The provider configuration string has the following syntax:

```
"PathToDll;SlotId;Pin"
```

- `PathToDll` is the path to driver library filename, which is provided by the manufacturer of the HSM, UBS token or smart card. Examples:
  - The SuisseID USB Tokens use *cvp11.dll*
  - The CardOS API from Atos (Siemens) uses *siecap11.dll*
  - The IBM 4758 cryptographic coprocessor uses *cryptoki.dll*
  - Devices from Aladdin Ltd. use *etpkcs11.dll*
- `SlotId` is optional, if it is not defined, it is searched for the first slot that contains a running token.
- `Pin` is optional, if it is not defined, the submission for the pin is activated via the pad of the token. If this is not supported by the token, the following error message is raised when signing: “Cannot access private key”.

Examples:

- `Provider = "\WINDOWS\system32\siecap11.dll;4;123456"`

#### Interoperability Support

The following cryptographic token interface (PKCS#11) products have been successfully tested:

- SafeNet Protect Server
- SafeNet Luna
- SafeNetAuthentication Client
- IBM OpenCrypTokI
- CryptoVision
- Siemens CardOS

#### Selecting a Certificate for Signing

The 3-Heights™ PDF to PDF/A Converter Shell offers different ways to select a certificate. The product tries the first of the following selection strategies, for which the required values have been specified by the user.

1. **Certificate fingerprint**
   Option -cfp
   - SHA1 fingerprint of the certificate. The fingerprint is 20 bytes long and can be specified in hexadecimal string representation, e.g. “b5 e4 5c 98 5a 7e 05 ff f4 c6 a3 45 13 48 0b c6 9d e4 5d f5”. In Windows certificate store this is called “Thumbprint”, if “Thumbprint algorithm” is “sha1”.
2. **Certificate Issuer and SerialNumber**
   Options -ci and -cno

- Certificate Issuer (e.g. “QV Schweiz CA”), in Windows certificate store this is called “Issued By”.
- Serial number of the certificate (hexadecimal string representation, e.g. “4c 05 58 fb”). This is a unique number assigned to the certificate by its issuer. In Windows certificate store this is the field called “Serial number” in the certificate’s “Details” tab.

3. **Certificate Name and optionally Issuer**
   Options -cn and -ci
   - Common Name of the certificate (e.g. “PDF Tools AG”), in Windows certificate store this is called “Issued To”.
   - Optional: Certificate Issuer (e.g. “QV Schweiz CA”), in Windows certificate store this is called “Issued By”.

#### Using PKCS#11 stores with missing issuer certificates

Some PKCS#11 devices contain the signing certificate only. However, in order to embed revocation information it is important, that the issuer certificates, i.e. the whole trust chain, is available as well.

On Windows, missing issuer certificates can be loaded from the Windows certificate store. So the missing certificates can be installed as follows:

1. Get the certificates of the trust chain. You can download them from the website of your certificate provider or do the following:
   (a) Sign a document and open the output in Adobe Acrobat
   (b) Go to “Signature Properties” and then view the signer’s certificate
   (c) Select a certificate of the trust chain
   (d) Export the certificate as “Certificate File” (extension .cer)
   (e) Do this for all certificates of the trust chain
2. Open the exported files by double clicking on them in the Windows Explorer
3. Click button “Install Certificate...”
4. Select “automatically select the certificate store based on the type of certificate” and finish import

### 4.5.2 Microsoft CryptoAPI Provider

Microsoft CryptoAPI (MS-CAPI, CAPI) offers access to the certificates stored in the Windows certificate store and other devices, such as USB tokens, with Windows integration. Microsoft CryptoAPI does not support some new cryptographic algorithms. Therefore it is recommended to use the PKCS#11 Provider if possible.

#### Configuration

**Provider** Option -cp

The provider configuration string has the following syntax:

```
"[ProviderType:]Provider[;PIN]"
```

The ProviderType and PIN are optional. The corresponding drivers must be installed on Windows.
Examples:

- `Provider = "Microsoft Base Cryptographic Provider v1.0"`
- `Provider = "Microsoft Strong Cryptographic Provider"`
- `Provider = "PROV_RSA_AES:Microsoft Enhanced RSA and AES Cryptographic Provider"`

The provider type PROV_RSA_AES supports the SHA-2 hash algorithms for signature validation (not signature creation). This provider type is recommended in order to validate signatures if no PKCS#11 device is available.
Optionally, when using an advanced certificate, the pin code can be passed as an additional, semi-column separated parameter. This does not work with qualified certificates, because they always require the pin code to be entered manually and every time. If the name of the provider is omitted, the default provider is used.
Examples, “123456” being the pin code:

- `Provider = "Microsoft Base Cryptographic Provider v1.0;123456"`
- `Provider = ";123456"`

**Certificate Store** Option -csn

The value for the certificate store depends on the OS. Supported values are: “CA”, “MY” and “ROOT”. For signature creation the default store “MY” is usually the right choice.

**Store Location** Option -csl

Either of the following store locations

- Local Machine
- Current User (default)

Usually personal certificates are stored in the current user location and company-wide certificates are stored under local machine.

The current user's store is only available, if the user profile has been loaded. This may not be the case in certain environments such as within an IIS web application or COM+ applications. Use the store of the Local Machine, if the user profile cannot be loaded.

Certificates in the store Local Machine are available to all users. However, in order to sign a document, you need access to the signing certificate's private key. The private key is protected by Windows ACLs and typically readable for Administrators only. Use the Microsoft Management Console (mmc.exe) in order to grant access to the private key for other users as follows: Add the Certificates Snap-in for the certificates on Local Machine. Right-click on the signing certificate, click on "All Tasks" and then "Manage Private Keys..." where you can set the permissions.

### Selecting a Certificate for Signing

First, the certificate store defined by the provider is used. Within the store the selection of the signing certificate works the same as with the PKCS#11 provider, which is described here: Selecting a Certificate for Signing

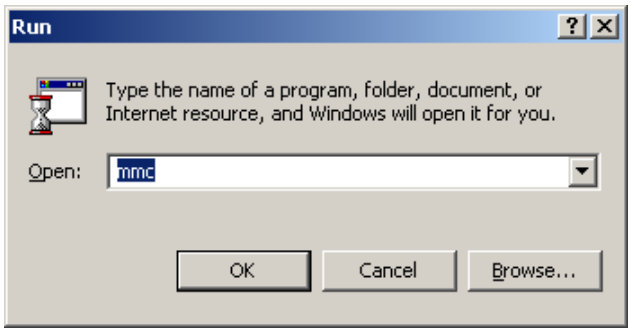
### Certificates

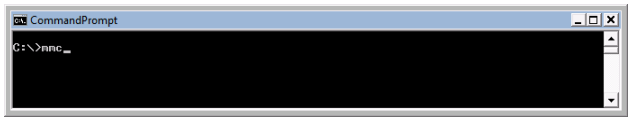
In order to sign a PDF document, a valid, existing certificate name must be provided and its private key must be available.

There are various ways to create or obtain a certificate. How this is done is not described in this document. This document describes the requirements for, and how to use the certificate.

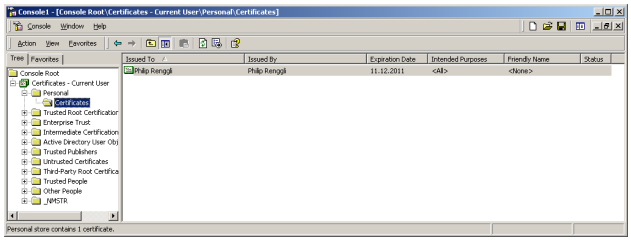
On the Windows operating system certificates can be listed by the Microsoft Management Console (MMC), which is provided by Windows. In order to see the certificates available on the system, do the following steps:

1. To launch the MMC, go to Start ->Run...->type "mmc", or start a Command Prompt and type "mmc".

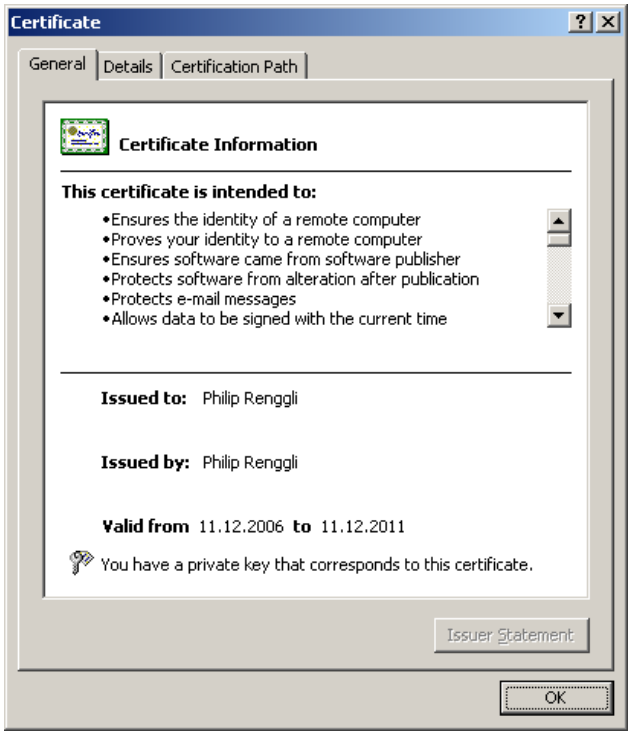


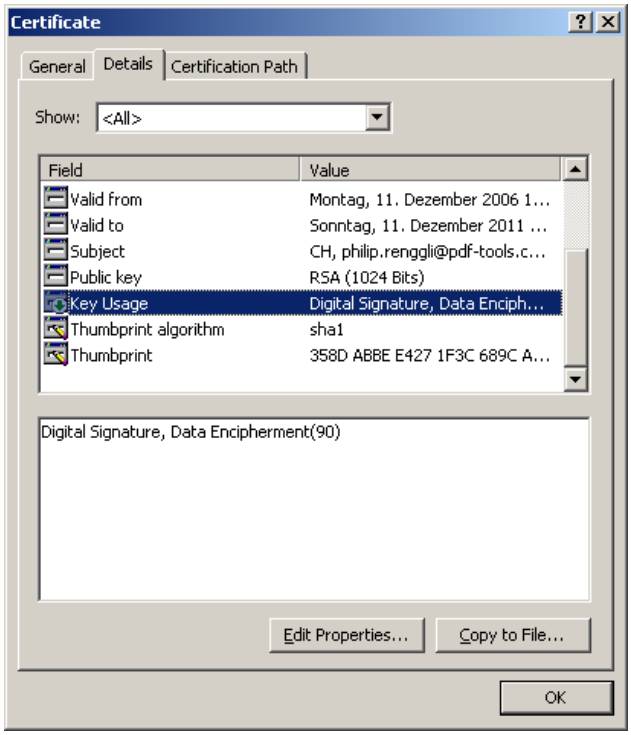

2. Under "File" ->"Add/Remove Snap-in"
3. Choose "Certificates" and click the "Add" button
4. In the next window choose to manage certificates for "My user account"
5. Click "Finish"
6. The certificate must be listed under the root "Certificates - Current User", for example as shown in the screenshot below:



7. Double-click the certificate to open. The certificate name corresponds to the value “Issued to:”.

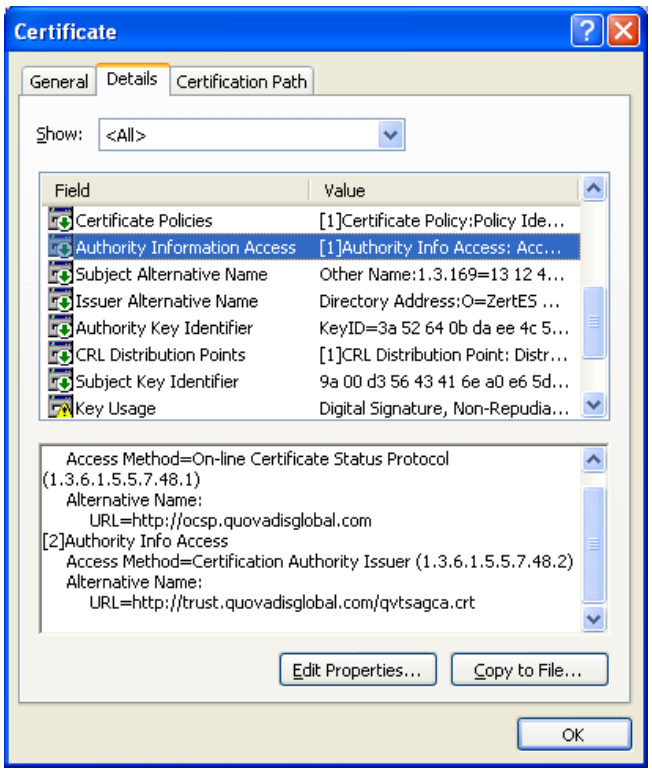

Certificate
General
Details
Certification Path
Certificate Information
This certificate is intended to:
•Ensures the identity of a remote computer
•Proves your identity to a remote computer
•Ensures software came from software publisher
•Protects software from alteration after publication
•Protects e-mail messages
•Allows data to be signed with the current time
Issued to: Philip Renggli
Issued by: Philip Renggli
Valid from 11.12.2006 to 11.12.2011
You have a private key that corresponds to this certificate.
Issuer Statement
OK

8. In the tab Detail of the certificate, there is a field named “Key Usage”. This field must contain the value “Digital Signature”. Additional values are optional, see also screenshot. You must have the private key that corresponds to this certificate.



### Qualified Certificates

A qualified certificate can be obtained from a certificate authority (CA). Besides the requirements listed in the previous chapter it has the additional requirement to contain the key “Authority Information Access” which contains the information about the OCSP server.




## 4.5.3 3-Heights™ Signature Creation and Validation Service

The 3-Heights™ Signature Creation and Validation Service provides HTTP protocol based remote access to cryptographic providers such as smartcards, USB tokens, and other cryptographic infrastructure such as HSMs.

Use of the 3-Heights™ Signature Creation and Validation Service provides the following advantages:

1. By means of this service the tokens can be hosted centrally and used by any client computer which has access to the service.
2. Cryptographic devices that can be used on Windows only can be made accessible to siging processes running on Non-Windows systems.
3. Cryptographic devices can be made accessible to processes running in non-interactive sessions. Many cryptographic devices must always be used in an interactive session for two reasons. First, the middleware requires the user to enter the pin interactively to create a qualified electronic signature. Second, USB tokens and smart cards are managed by Windows such that the device is available only to the user currently using the computer’s console. Therefore, services, remotely logged in users and applications running in locked sessions have no access to the device.

Note that this is a separate product and this chapter describes its usage with the 3-Heights™ PDF to PDF/A Converter Shell only. For more information on the 3-Heights™ Signature Creation and Validation Service and installation instructions, please refer to its separate user manual.

### Configuration

**Provider** Option -cp

The provider configuration string has the following syntax:

`"http://server.mydomain.com:<port>/<token>;<password>"`

Where:

- `server.mydomain.com` is the hostname of the server
- `<port>` is optional, port of the server.
- `<token>` the ID of the token.

- `<password>` password of the token.

Examples:

- `Provider = "http://server.mydomain.com:8080/0001;pass01"`

A more detailed description can be found in the user manual of the 3-Heights™ Signature Creation and Validation Service.

### Selecting a Certificate for Signing

Selection of the signing certificate works the same as if the token was used directly: Selecting a Certificate for Signing.

### 4.5.4 SwissSign Personal Signing Service

**Provider** Option -cp

The provider configuration string contains the URL to the service endpoint.

**Provider Configuration**

The provider can be configured using provider session properties.

There are two types of properties:

- *String Properties:*
  String properties are set using option -cps.
- *File Properties:*
  File properties are set using option -cpf.

| Name | Type | Required | Value |
|---|---|---|---|
| **Identity** | String | required | The identity of your signing certificate. Example: `My Company:Signing Cert 1` |
| **DSSProfile** | String | required | `http://dss.swisssign.net/dss/profile/pades/1.0` |
| **SSLClientCertificate** | File | required | SSL client certificate in PKCS#12 Format (.p12, .pfx). File must contain the certificate itself, all certificates of the trust chain and the private key. |
| **SSLClientCertificatePassword** | String | optional | Password to decrypt the private key of the SLL client certificate. |
| **SSLServerCertificate** | File | recommended | Certificate of the server or its issuer (CA) certificate in DER Format (.der, .cer) Note: If this property is not set, the server certificate is not verified at all! |
| **RequestID** | String | recommended | Any string that can be used to track the request. Example: An UUID like `AE57F021-C0EB-4AE0-8E5E-67FB93E5BC7F` |

**Signature Configuration**

The signature can be customized using standard options of the 3-Heights™ PDF to PDF/A Converter Shell.

| Description | Required | Value | Setting |
|---|---|---|---|
| **Common Name** | required | The name of the signer should be set[5]. | Option -cn. |
| **Time-stamp** | optional | `"urn:ietf:rfc:3161"` | Option -tsu |
| **Revocation Info** | optional | `true` to embed OCSP responses or CRL. | Option -co |
| **Visual Appearance** | optional | See separate chapter on creating a visual appearance. | |

**Proxy Configuration**

If a proxy is used for the connection to the service, see chapter How to Use a Proxy for more information.

[5]This parameter is not used for certificate selection, but for the signature appearance and signature description in the PDF only.

### 4.5.5 SwissSign SuisseID Signing Service

In order to use the SuisseID Signing Service, please contact Swiss Post Solutions AG (suisseid@post.ch) to obtain access credentials. Prior to invoking the SuisseID Signing Service, user authentication via the SuisseID Identity Provider (IDP) is a pre-requisite. So the calling application must integrate via SAML (e.g. SuisseID SDK) with the SuisseID Identity Provider. The IDP issues SAML tokens upon successful user authentication.

Note that the name of the signature should be the signer's name (e.g. "<givenname> <surname>"). The signer's name can be retrieved for the SAML token as the IDP provides this as qualified attributes (yellowid verified).

**Provider** Option -cp

The provider configuration string contains the URL to the service Endpoint.

**Provider Configuration**

The provider can be configured using provider session properties.

There are two types of properties:

- *String Properties:*
  String properties are set using option -cps.
- *File Properties:*
  File properties are set using option -cpf.

| Name | Type | Required | Value |
|---|---|---|---|
| **SAMLToken** | File | required | SAML token issued by the SuisseID Identity Provider (IDP).<br>Example: `C:\temp\my-saml.xml` |
| **SSLClientCertificate** | File | required | SSL client certificate in PKCS#12 Format (.p12, .pfx).<br>File must contain the certificate itself, all certificates of the trust chain and the private key. |
| **SSLClientCertificatePassword** | String | optional | Password to decrypt the private key of the SLL client certificate. |
| **SSLServerCertificate** | File | recommended | Certificate of the server or its issuer (CA) certificate in DER Format (.der, .cer)<br>Note: If this property is not set, the server certificate is not verified at all! |

**Signature Configuration**

The signature can be customized using standard options.

| Description | Required | Value | Setting |
|---|---|---|---|
| **Common Name** | required | The name of the signer should be set[6]. | Option -cn. |
| **Time-stamp** | optional | `"urn:ietf:rfc:3161"` | Option -tsu |
| **Revocation Info** | optional | `true` to embed OCSP responses or CRL. | Option -co |
| **Visual Appearance** | optional | See separate chapter on creating a visual appearance. | |

**Proxy Configuration**

If a proxy is used for the connection to the service, see chapter How to Use a Proxy for more information.

[6]This parameter is not used for certificate selection, but for the signature appearance and signature description in the PDF only.

### 4.5.6 QuoVadis sealsign

**Provider** Option -cp

The provider configuration string contains the URL to the QuoVadis sealsign service. For example:
`"http://services.sealsignportal.com:18080/sealsign/ws/BrokerClient"`

**Provider Configuration**

The provider can be configured using provider session properties that can be set using the options -cps or -cpf.

| Name | Type | Required | Value |
|---|---|---|---|
| **Identity** | String | required | The account ID is the unique name of the account specified on the server.<br>Example: `Rigora` |
| **Profile** | String | required | The profile identifies the signature specifications by a unique name.<br>Example: `Default` |
| **secret** | String | required | The secret is the password which secures the access to the account.<br>Example: `NeE=EKEd33FeCk70` |
| **clientId** | String | optional | A client ID can be used to help separating access and creating better statistics. If specified in the account configuration it is necessary to provide this value.<br>Example: `3949-4929-3179-2818` |
| **pin** | String | required | The PIN code is required to activate the signing key.<br>Example: `123456` |
| **MessageDigestAlgorithm** | String | optional | The message digest algorithm to use. Note that the supported algorithms depend on the provider.<br>Default: `SHA-256`<br>Alternatives: `SHA-1`, `SHA-384`, `SHA-512`, `RIPEMD-160`, `RIPEMD-256` |

**Signature Configuration**

The signature can be customized using standard options.

| Description | Required | Value | Setting |
|---|---|---|---|
| **Common Name** | required | The name of the signer should be set[7]. | Option -cn. |
| **Time-stamp** | - | Not available. | |
| **Revocation Info** | optional | `true` to embed OCSP responses or CRL. | Option -co |
| **Visual Appearance** | optional | See separate chapter on creating a visual appearance. | |

**Proxy Configuration**

If a proxy is used for the connection to the service, see chapter How to Use a Proxy for more information.

[7] This parameter is not used for certificate selection, but for the signature appearance and signature description in the PDF only.

### 4.5.7 Swisscom All-in Signing Service

#### General Properties

To use the signature service, the following general properties have to be set:

| Description | Required | Value | Setting |
|---|---|---|---|
| **Common Name** | required | Name of the signer[8]. | Option -cn |
| **Provider** | required | The service endpoint URL of the REST service. Example: `https://ais.swisscom.com/AIS-Server/rs/v1.0/sign` | Option -cp |
| **Time-stamp** | optional | `"urn:ietf:rfc:3161"` | Option -tsu |
| **Revocation Info** | optional | `true` to embed OCSP responses | Option -co |

If a proxy is used for the connection to the service, see chapter How to Use a Proxy for more information.

#### Provider Session Properties

In addition to the general properties, a few provider specific session properties have to be set.

There are two types of properties:

- *String Properties:*
  String properties are set using option -cps.
- *File Properties:*
  File properties are set using option -cpf.

| Name | Type | Required | Value |
|---|---|---|---|
| **DSSProfile** | String | required | `http://ais.swisscom.ch/1.0` |
| **SSLClientCertificate** | File | required | SSL client certificate in PKCS#12 Format (.p12, .pfx). File must contain the certificate itself, all certificates of the trust chain and the private key. |
| **SSLClientCertificatePassword** | String | optional | Password to decrypt the private key of the SLL client certificate. |
| **SSLServerCertificate** | File | recommended | Certificate of the server or its issuer (CA) certificate in DER Format (.der, .cer) Note: If this property is not set, the server certificate is not verified at all! |
| **Identity** | String | required | The identity string as provided by Swisscom: `<customer name>:<key identity>` |
| **RequestID** | String | recommended | Any string that can be used to track the request. Example: An UUID like `AE57F021-C0EB-4AE0-8E5E-67FB93E5BC7F` |

[8]This parameter is not used for certificate selection, but for the signature appearance and signature description in the PDF only.

#### On-Demand Certificates

To request an on-demand certificate, the following additional property has to be set:

| Name | Type | Required | Value |
|---|---|---|---|
| **SwisscomAllInOnDemandDN** | String | required | The requested distinguished name.<br>Example: `cn=Hans Muster,o=ACME,c=CH` |

#### Step-Up Authorization using Mobile-ID

To use the step-up authorization, the following additional properties have to be set:

| Name | Type | Required | Value |
|---|---|---|---|
| **SwisscomAllInMSISDN** | String | required | Mobile phone number.<br>Example: `+41798765432` |
| **SwisscomAllInMessage** | String | required | The message to be displayed on the mobile phone.<br>Example: `Pipapo halolu.` |
| **SwisscomAllInLanguage** | String | required | The language of the message.<br>Example: `DE` |

Those properties have to comply with the Swisscom Mobile-ID specification.

## 4.6 How to Create Digital Signatures

This chapter describes the steps that are required to create different types of digital signatures. A good introductory example can be found in the chapter Digital Signatures.

### 4.6.1 How to Create a PAdES LTV Signature

In order to create a PAdES LTV signature, the following is required:

1. *An advanced or qualified signing certificate.*
   For requirements and preparation steps see the sample in chapter Digital Signatures. Make sure the store of your cryptographic provider contains all certificates of the trust chain, including the root certificate.
2. *Embed revocation information.*
   Do not use the option -co.
3. *Add a Time-stamp.*
   Use the option -tsu.
4. *Proper error handling.*
   A proper error handling is crucial in order to ensure the creation of correctly signed documents. The output document was signed successfully, if and only if the conversion error mask (option -cem) includes the flags 1024 (Failed to create signature) and 2048 (Failed to add OCSP, CRL or TSP to signature.) and 3-Heights™ PDF to PDF/A Converter Shell returns code 0 (success).

### 4.6.2 How to Create a Visual Appearance of a Signature

Each signature may have a visual appearance on a page of the document. The visual appearance is optional and has no effect on the validity of the signature. Because of this and because a visual appearance may cover important content of the page, many applications choose to create an invisible signature.

By default, the 3-Heights™ PDF to PDF/A Converter Shell creates an appearance in the lower left corner of the last page, which looks as shown below:

#### How to Create an Invisible Signature

Invisible signatures have no visual appearance and can be created by setting an empty rectangle using the Option -ar:

```
-ar 0 0 0 0
```

#### How to Create a Visual Appearance

Different properties of the visual appearance can be specified.

**Page and Position**
: See options -ap and -ar.

**Text**
: Two text fragments can be set using two different fonts and font sizes, see options -at1, -at2, -af1 and -af2.

**Background image**
: See options -abg.

### 4.6.3 Miscellaneous

#### Caching of CRLs, OCSP and TSP Reponses

In order to improve the speed when mass signing, the 3-Heights™ PDF to PDF/A Converter Shell provides a caching algorithm to store CRL (Certificate Revocation List), OCSP (Online Certificate Status Protocol), TSP (Time-stamp Protocol) and data from signature services. This data is usually valid over period of time that is defined by the protocol, which is normally at least 24 hours. Caching improves the speed, because there are situations when the server does not need to be contacted for every digital signature. The following caches are stored automatically by the 3-Heights™ PDF to PDF/A Converter Shell at the indicated locations within the directory for temporary files:

| | |
|---|---|
| OCSP responses: | *ocsp/server-hash.der* |
| CRL: | *crl/server.der* |
| TSP responses[9] | *tsp/server.der* |
| Service data: | *sig/hash.bin* |

The caches can be cleared by deleting the files. Usage of the caches can be deactivated by setting the option -nc. The files are updated if the current date and time exceeds the “next update” field in the OCSP or CRL response respectively or the cached data was downloaded more than 24 hours ago.

The directory for temporary files is determined as follows. The product checks for the existence of environment variables in the following order and uses the first path found:

[9] The sizes of the TSP responses are cached only. Cached TSP responses cannot be embedded but used for the computation of the signature length only.

- *Windows:*
  1. The path specified by the *TMP* environment variable.
  2. The path specified by the *TEMP* environment variable.
  3. The path specified by the *USERPROFILE* environment variable.
  4. The Windows directory.
- *Unix:*
  1. The path specified by the *PDFTMPDIR* environment variable.
  2. The path specified by the *TMP* environment variable.
  3. The */tmp* directory.

### How to Use a Proxy

The 3-Heights™ PDF to PDF/A Converter Shell can use a proxy server for all communication to remote servers, e.g. to download CRL or for communication to a signature service. The proxy server can be configured using the provider session property *Proxy*. The property's value must be a string with the following syntax:

```
http[s]://<user:password>@host:<port>
```

Where:

- `http` / `https`: Protocol for connection to proxy.
- `user:password` (optional): Credentials for connection to proxy (basic authorization).
- `host`: Hostname of proxy.
- `port`: Port for connection to proxy.

For SSL connections, e.g. to a signature service, the proxy must allow the HTTP CONNECT request to the signature service.

**Example:** Configuration of a proxy server that is called `myproxy` and accepts HTTP connections on port `8080`.

```
-cps "Proxy" "http://myproxy:8080"
```

### Configuration of Proxy Server and Firewall

For the application of a Time-stamp or online verification of certificates, the signature software requires access to the server of the certificates' issuer (e. g. http://ocsp.quovadisglobal.com or http://platinum-qualified-g2.ocsp.swisssign.net/) via HTTP. The URL for verification is stored in the certificate; the URL for Time-stamp services is provided by the issuer. In case these functions are not configured, no access is required.

In organizations where a web proxy is in used, it must be ensured that the required MIME types are supported. These are:

**OCSP**

- `application/ocsp-request`
- `application/ocsp-response`

**Time-stamp**

- `application/timestamp-query`
- `application/timestamp-reply`

**Signature services**

- Signature service specific MIME types.

## 4.7 How to Validate Digital Signatures

### Validation of a Qualified Electronic Signature

There are basically three items that need to be validated:

1. Trust Chain
2. Revocation Information (optional)
3. Time-stamp (optional)

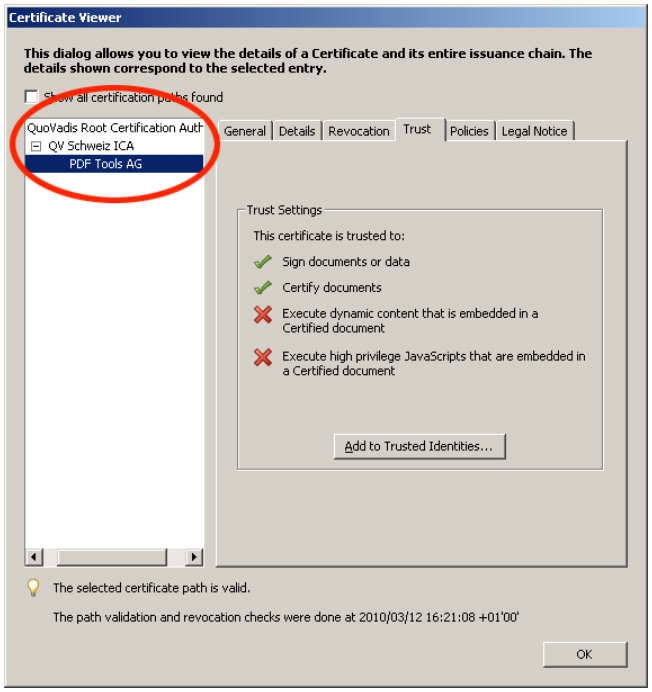
Validation can be in different ways, e.g. Adobe Acrobat, from which the screenshots below are taken.

#### Trust Chain

Before the trust chain can be validated, ensure the root certificate is trusted. There are different ways to add a certificate as trusted root certificate. The best way on Windows is this:

1. Retrieve a copy of the certificate containing a public key. This can be done be requesting it from the issuer (your CA) or by exporting it from an existing signature to a file (*CertExchange.cer*). Ensure you are not installing a malicious certificate!
2. Add the certificate to the trusted root certificates. If you have the certificate available as file, you can simply double-click it to install it.

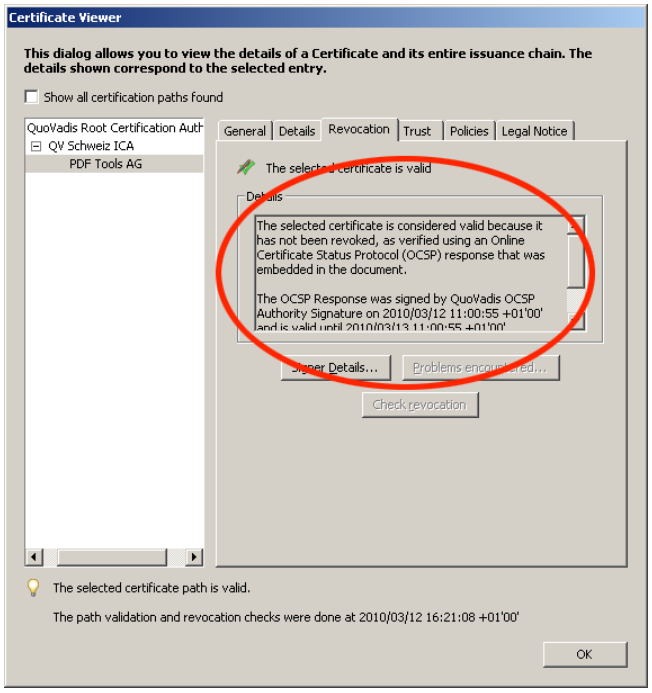
After that you can validate the signature, e.g. by open the PDF document in Adobe Acrobat, right-click the signature and select “Validate”, then select “Properties” and select the tab “Trust”. There the certificate should be trusted to “sign documents or data”.


Certificate Viewer
This dialog allows you to view the details of a Certificate and its entire issuance chain. The details shown correspond to the selected entry.
Show all certification paths found
QuoVadis Root Certification Auth
QV Schweiz ICA
PDF Tools AG
General
Details
Revocation
Trust
Policies
Legal Notice
Trust Settings
This certificate is trusted to:
Sign documents or data
Certify documents
Execute dynamic content that is embedded in a Certified document
Execute high privilege JavaScripts that are embedded in a Certified document
Add to Trusted Identities...
The selected certificate path is valid.
The path validation and revocation checks were done at 2010/03/12 16:21:08 +01'00'
OK

### Revocation Information

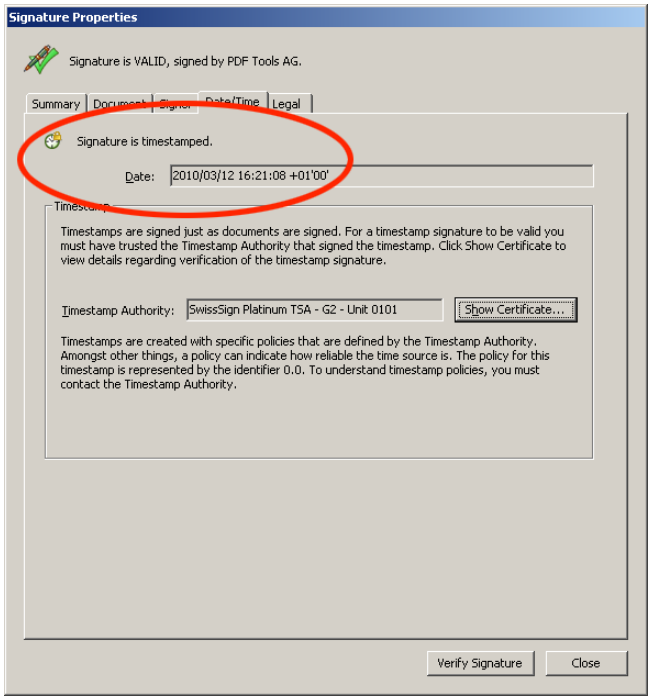
An OCSP response or CRL must be available. This is shown in the tab “Revocation”. The details should mention that “the certificate is considered *valid*”.

The presence of revocation information must be checked for the signing certificate and all certificates of its trust chain except for the root certificate.


Certificate Viewer
This dialog allows you to view the details of a Certificate and its entire issuance chain. The details shown correspond to the selected entry.
Show all certification paths found
QuoVadis Root Certification Auth
QV Schweiz ICA
PDF Tools AG
General
Details
Revocation
Trust
Policies
Legal Notice
The selected certificate is valid
Details
The selected certificate is considered valid because it has not been revoked, as verified using an Online Certificate Status Protocol (OCSP) response that was embedded in the document.
The OCSP Response was signed by QuoVadis OCSP Authority Signature on 2010/03/12 11:00:55 +01'00'
Signer Details...
Problems encountered...
Check revocation
The selected certificate path is valid.
The path validation and revocation checks were done at 2010/03/12 16:21:08 +01'00'
OK

### Time-stamp

The signature can optionally contain a Time-stamp. This is shown in the tab “Date/Time”. The certificate of the Time-stamp server must also be trusted, i.e. its trust chain should be validated as described in the section Trust Chain above.


Signature Properties
Signature is VALID, signed by PDF Tools AG.
Summary
Legal
Signature is timestamped.
Date:
2010/03/12 16:21:08 +01'00'
Timestamps are signed just as documents are signed. For a timestamp signature to be valid you must have trusted the Timestamp Authority that signed the timestamp. Click Show Certificate to view details regarding verification of the timestamp signature.
Timestamp Authority:
SwissSign Platinum TSA - G2 - Unit 0101
Show Certificate...
Timestamps are created with specific policies that are defined by the Timestamp Authority. Amongst other things, a policy can indicate how reliable the time source is. The policy for this timestamp is represented by the identifier 0.0. To understand timestamp policies, you must contact the Timestamp Authority.
Verify Signature
Close

### Validation of a PAdES LTV Signature

Verifying if a signature conforms to the PAdES LTV standard is similar to validating a Qualified Electronic Signature.

The following must be checked:

1. Trust Chain
2. Revocation information
3. Time-stamp
4. LTV expiration date
5. Other PAdES Requirements

#### Trust Chain

Trust chain validation works the same as for validating Qualified Electronic Signatures.

#### Revocation information

Revocation information (OCPS response or CRL) must be valid and embedded into the signature. In the details, verify that the revocation check was performed using data that was "was *embedded in the signature*". Revocation information that "was contained in the *local cache*" or "was requested *online*" is not embedded into the signature and does not meet PAdES LTV requirements.

#### Time-stamp

A Time-stamp must be embedded and validated as described for validating Qualified Electronic Signatures. If a document contains multiple Time-stamps, all but the latest one must contain revocation information.

#### LTV expiration date

The long term validation ability expires with the expiration of the signing certificate of the latest Time-stamp. The life-time of the protection can be further extended beyond the life-of the last Time-stamp applied by adding further DSS information to validate the previous last Time-stamp along with a new Time-stamp.

#### Other PAdES Requirements

Certain other PAdES requirements, such as requirements on the PKCS#7 CMS, cannot be validated using Adobe Acrobat. For this, use the 3-Heights™ PDF Security API for validation.

# 5 Reference Manual

## 5.1 General Settings

#### -cl: Set Conformance

Set the PDF/A conformance level. Support parameters are: pdfa-1b, pdfa-1a, pdfa-2b, pdfa-2u, pdfa-2a, pdfa-3b, pdfa-3u, pdfa-3a. The default value is pdfa-2b.

Some files cannot be converted to the compliance requested. The 3-Heights™ PDF to PDF/A Converter Shell can detect this and up- or downgrade the compliance automatically. Use the option `-cem` to prevent automatic up- or downgrades.

The following example will set the conformance level to PDF/A-2u:

**Example**

```
-cl pdfa-2u
```

### -cem: Mask Conversion Errors

The conversion error mask defines which operations and conditions are not allowed and consequently cause a conversion error (return value 5).

0 None (never return a conversion error)

1 Set this flag to prevent automatic upgrades from PDF/A-1 to PDF/A-2. When converting a file to PDF/A-1, transparency needs to be removed. This may cause significant visual differences. Therefore, the default behavior of the 3-Heights™ PDF to PDF/A Converter Shell is to convert the file to PDF/A-2.

2 Set this flag to prevent automatic downgrades of the conformance level, e.g. from PDF/A-1a to PDF/A-1b. If this flag is not set, the level is downgraded under the following conditions:

- Downgrade to level B: If a file contains text that is not extractable (i.e. missing ToUnicode information).
  Example: Downgrade PDF/A-2u to PDF/A-2b.
- Downgrade to level U (PDF/A-2 and PDF/A-3) or B (PDF/A-1): Level A requires logical structure information ("tagging") information, so if a file contains no such information, its level is downgraded.
  Logical structure information in a PDF defines the structure of content, such as titles, paragraphs, figures, reading order, tables or articles. Logical structure elements can be "tagged" with descriptions or alternative text. "Tagging" allows the contents of an image to be described to the visually impaired.
  It is not possible for the 3-Heights™ PDF to PDF/A Converter Shell to add meaningful tagging information. Adding tagging information without prior knowledge about the input file's structure and content is neither possible nor allowed by the PDF/A standard. For that reason, the conformance level is automatically downgraded to level B or U.
  Example: Downgrade PDF/A-1a to PDF/A-1b.

4 (default) Visual differences in output file.

8 Resolve name collisions of colorants (PDF/A-2 and PDF/A-3 only).

16 (default) Remove optional content groups (layers) (PDF/A-1 only).

32 (default) Remove transparency (PDF/A-1 only).

64 (default) Remove embedded files.

128 (default) Remove non convertible XMP metadata.

256 (default) Error during linearization of output file.

512 Conversion of signed document forced removal of signatures.

1024 (default) Failed to create signature.

2048 Failed to add revocation information (OCSP or CRL) or TSP to signature.

4096 (default) The input document is corrupt and should be repaired. The errors encountered are printed to the log file. Some errors can be repaired, but it is crucial to review the output file and perform the post analysis.

8192 (default) OCR error occurred.

16384 Font substituted.

Add up all operations are conditions to define the conversion mask. The default is 13812. In order to accept all conversion errors, set the mask to 0.

```
-cem 0
```

### -fd: Set font directory

In order to create a valid PDF/A all fonts must be embedded. If the input file contains a font which is not embedded, the font folder is searched for a font with the same name. If such a font is found, the font is embedded. If no valid font directory is added, the default font directories are used. The location of these directories depends on the operating system:

- Windows: %SystemRoot%\Fonts and directory "Fonts", which must be a direct sub-directory of where the main DLL or executable resides.
- Mac: /System/Library/Fonts and /Library/Fonts
- Unix: $PDFFONTDIR or /usr/lib/X11/fonts/Type1

See chapter Fonts for more information on the font directories and font handling of the 3-Heights™ PDF to PDF/A Converter Shell in general.

If replacement fonts shall be taken from another location, this location can be set using the switch -fd:

```
-fd C:\MyFonts
```

### -id: Set value in the document information dictionary

Set or get the value of an info entry key. Examples for keys are "Author", "Subject", "Title", "Producer" or custom attributes.

### -ma: Analyze the Input File

Analyze the input file and verify if it meets a certain compliance level. In order to get a report either the option -rs or -rd can be used in combination, otherwise only the return code it set.

```
pdf2pdf -ma -rs input.pdf output.pdf
The document contains fonts without embedded font programs or encoding information
(CMAPs). The documents meta data is either missing or inconsistent or corrupt.
```

An output file name must be provided, since the output name also specify the name of the log file which is generated. However no output PDF document is created. The analysis is equal to the analysis using the 3-Heights™ PDF Validator and validating against PDF/A.

### -mc: Force Conversion even if there Are Analysis Errors

Setting this option forces the conversion even if there is a problem with the input file. A conversion to PDF/A can fail if the document stands in conflict with one of the following issues:

- Non-conformance with the PDF Reference
- Layers
- Transparency
- Missing or ambiguous annotation and form field appearances
- Unknown annotation types (optional)
- Multimedia annotations (optional)

Under the following circumstances the conversion is automatically downgraded to PDF/A-1b, (in case PDF/A-1a was selected).

- Missing Unicode information
- Missing logical structure

### -mp: Post-Analyze the Result

Analyze the created output file directly after the conversion to verify it meets the required compliance level. In order to get a report either the option -rs or -rd can be used in combination, otherwise only the return code it set. The post analysis is only executed if the conversion was successful. This switch is ignored if the switch -ma (Analysis only) is set. The post analysis can detect errors in the created output file that could not be predicted based on the analysis of the input file nor could they be detected during the conversion, because the conversion also depends on the input parameters (such as ICC profiles).

```
-mp -rs
```

The post-analysis is equal to the analysis using the 3-Heights™ PDF Validator and validating against PDF/A.

### -p: Read an Encrypted PDF File

When the input PDF file is encrypted and has a user password set, (the password to open the PDF) the password can be provided with the option -p. If for example the user password were “userpwd”, then the command would look like this:

```
-p userpwd
```

When a PDF is encrypted and the user password is not provided or is incorrect, pdf2pdf cannot decrypt and read the file. Instead it will generate the following error message:

```
0x80410112 - E - The authentication failed due to a wrong password.
Couldn't open input file input.pdf.
```

### -ax: Add XMP Metadata

Add XMP meta data from a file. Providing a path that does not exist or an invalid XMP file results in return code 3.

```
-ax metadata.txt
```

### -ow: Optimize for the Web

Add so called linearization tags to the document. A linearized document has a slightly larger file size than a non-linearized file, and provides the following features (among others):

- When a document is opened through a PDF viewing application plug-in for an Internet browser, the first page can be viewed without downloading the entire PDF file.
- When another page is requested by the user, that page is displayed as quickly as possible and incrementally as data arrives, without downloading the entire PDF file.

### -q: Image Quality

Set or get the image quality index for images that use a prohibited lossy compression type are must be recompressed.

*Example:* JPX is not allowed in PDF/A-1. If a PDF contains a JPX compressed image, its compression type must be altered. Thus the 3-Heights™ PDF to PDF/A Converter Shell converts it to an image with regular JPEG compression and the image quality as defined by this switch.

Supported values are 1 to 100. A higher value means better visual quality at the cost of a larger file size. Recommended values range from 70 to 90. The default value is 75.

### -rd: Report Conformance Violations in Detail

This option lists all conformance violations per page. Each violation is listed with a page number (page 0 = document level), error number, a description and a counter of how many times the error occurs. The option provides more detailed information than the summary (Options -rs). This option can be used in combination with -mp or -ma.

**Example**

```
pdf2pdf -ma -rd input.pdf output.pdf
0, 0x80410604, "The key Metadata is required but missing.", 1
0, 0x80410604, "The key MarkInfo is required but missing.", 1
1, 0x00418704, "The font Arial-BoldMT must be embedded.", 1
1, 0x00418704, "The font Arial-BlackItalic must be embedded.", 1
1, 0x83410612, "The document does not conform to the requested standard.", 1
```

### -rs: Report Conformance Violations Summary

This option gives a summary of all conformance violations. If any of the following violations is detected at least once, it is reported (once). This option provides less detailed information than the detailed list per page (Option -rd).

- This option can be used in combination with -mp or -ma
- The file format (header, trailer, objects, xref, streams) is corrupted.
- The document doesn't conform to the PDF reference (missing required entries, wrong value types, etc.).
- The file is encrypted and the password was not provided.
- The document contains device-specific color spaces.
- The document contains illegal rendering hints (unknown intents, interpolation, transfer and halftone functions).
- The document contains alternate information (images).
- The document contains embedded PostScript code.
- The document contains references to external content (reference XObjects, file attachments, OPI).
- The document contains fonts without embedded font programs or encoding information (CMAPs).
- The document contains fonts without appropriate character to Unicode mapping information (ToUnicode maps).
- The document contains transparency.
- The document contains unknown annotation types.
- The document contains multimedia annotations (sound, movies).
- The document contains hidden, invisible, non-viewable or non-printable annotations.
- The document contains annotations or form fields with ambiguous or without appropriate appearances.
- The document contains actions types other than for navigation (launch, JavaScript, ResetForm, etc.).
- The document's meta data is either missing or inconsistent or corrupt.
- The document doesn't provide appropriate logical structure information.
- The document contains optional content (layers).

### -cff: Embed Type 1 fonts as CFF

Convert Type1 (PostScript) fonts to Compact Font Format before embedding. This reduces the file size. This affects the embedding of fonts only, existing Type1 fonts of the input document will not be converted.

### -uf: Update the fonts' Unicodes

Update the fonts' Unicodes as specified by file. The file must contain the mapping of character codes to Unicodes for specific fonts.

### -ef: Add embedded file

`-ef "file"` is the same as `-af ";;;file"`.

#### -af: Add associated file

Add a file to the document's embedded files. For PDF/A-3, the embedded file is associated with an object of the document, i.e. it is an associated file. The file is embedded as-is. Embedding files is not allowed for PDF/A-1 and restricted to PDF/A compliant files for PDF/A-2. The argument is of the form "`i;r;m;d;f`" where:

`i` The object to associate the embedded file with. -1 for none, 0 for document, number greater than 0 for respective page. Default: 0 for PDF/A-3 and -1 otherwise.

`r` The relationship of the embedded file to the object associated, PDF/A-3 only. Allowed values are "Source", "Data", "Alternative", "Supplement", and "Unspecified". Default: "Unspecified".

`m` Mime-type of the embedded file. Default: "application/octet-stream". Other common values are "application/pdf", "application/xml", or "application/msword".

`d` A description of the embedded file. This is presented to the user when viewing the list of embedded files.

`f` The path (or URL) to the file to be embedded.

**Example**

```
pdf2pdf -cl pdfa-3a -af "0;Source;application/msword;The source
    document;input.doc" input.pdf output.pdf
```

#### -az: Add a ZUGFeRD XML invoice file.

Add a ZUGFeRD XML invoice file. Note that this requires the compliance to be set to PDF/A-3. If the specified ZUGFeRD XML invoice file cannot be added during conversion, the following conversion error event is generated: *Remove embedded files.*

#### -v: Verbose Mode

This option turns on the verbose mode.

#### -lk: Set License Key

Pass a license key to the application at runtime instead of installing it on the system.

## 5.2 Processing Files in a Directory

Wildcards return a list of existing files. If you would like to convert all files in a directory to individual PDF documents, it is required to use a variable to name the output files. Here is an example for the FOR-command of the CMD-shell. It converts all TIFF files to PDFs with the same name and the extension ".pdf", in the same folder:

```
for %i in (*.pdf) do pdf2pdf -v %i %~ni.pdf
```

Of course, one can adjust the paths, or use a different output name:

```
for %i in (.\input\*.pdf) do pdf2pdf %i .\output\new_%~ni.pdf
```

For additional help to the FOR command, use the command (Windows):

```
for /?
```

Note, that variables used in a batch file require two leading "%" instead of one.

## 5.3 Color Profiles

See the dedicated section Color Profiles for more information on the topic.

### -cs: ICC Profile for Device-Specific Color Spaces

This ICC profile represents the color profile of the scanner. It is required if a color space is used that is different from color ICC profile of the output intent. Initially there is a default color profile for RGB (sRGB) and CMYK (USWebCoatedSWOP.icc) defined in the 3-Heights™ PDF to PDF/A Converter. This switch can be used to set both, the RGB and the CMYK color profile. If an RGB color profile is passed as argument it is set as new RGB color space, if a CMYK color is provided, it is set as new CMYK color space, if an invalid file is provided, it results in error code 3. To set the color profile for both color spaces, use the switch -cl twice. The following command sets the standard sRGB color profile as color space:

```
-cs "C:\WINNT\system32\spool\drivers\color\sRGB Color Space Profile.icm"
```

If a required color space profile is not available, a conversion error is generated.

### -oi: ICC Profile for Output Intent

The ICC profile for the output intent describes the color profile of the device (monitor or display). An output intent is required for PDF/A compatibility when converting images that do not have an embedded color profile. If no output intent is specified, a default color profile is embedded. The default color profile is the *sRGB Color Space Profile.icm*. If the input document already contains an output intent, the existing output intent is kept. Providing a path that does not exist or an invalid ICC color profile file results in return code 3.

```
-oi "C:\WINNT\system32\spool\drivers\color\sRGB Color Space Profile.icm"
```

## 5.4 Digital Signatures

For more information on digital signatures in general, see section Digital Signatures. For more information on how to create digital signatures, see section How to Create Digital Signatures.

### -ap: Signature Page Number

Set the page number of where the visual appearance of the digital signature should be placed. The default is the last page. The last page can also be set using -1 as argument.

### -ar: Signature Annotation Rectangle

This option allows positioning the digital signature annotation. The default location is in the lower left corner. The units are PDF points (A4 = 595x842 points, Letter = 612x792 points).

**Example:** create a 200 by 60 points rectangle in the upper left corner of an A4 page

```
-cn "..." -ar 10 770 200 60
```

In order to create an invisible signature use the following rectangle:

```
-ar 0 0 0 0
```

### -cn: Certificate Name (Subject)

In order to sign a PDF document, a valid, existing certificate name must be provided. Consult the chapter “Certificates” to learn more about certificates. The name of a certificate is to be provided as parameter to the -cn switch to digitally sign a PDF document as shown in the command below: This property can be used to select the signer certificate for signing (see description of Cryptographic Provider in use).

**Example:** Sign the document.

```
-cn "Philip Renggli"
```

The signature is added on the last page of the signed document.

### -cr: Signature Reason

Add a descriptive text about the reason why the document was signed.

**Example:** Sign the document and add a reason text.

```
-cn "Philip Renggli" -cr "I reviewed the document"
```

The signature of the resulting output looks as shown below:

### -cci: Signer contact info

Add a descriptive text as signer contact info, e.g. a phone number. This enables a recipient to contact the signer to verify the signature. This is not required in order to create a valid signature.

### -ca: Abort Conversion if Document Is Signed

If -ca is set and the input document is not PDF/A compliant (i.e. a conversion is required) and contains a digital signature, the conversion is aborted and the error code 11 is returned.

If -ca is not set and the input document is not PDF/A compliant, all signatures (including MDP signature) and signature appearances of the input document are removed.

If the input document is PDF/A compliant, the document is not converted and the existing signatures remain. Optionally an additional signature can be applied.

### -ci: Certificate Issuer

The issuer of the certificate. The “Certificate Issuer” corresponds to the common name (CN) of the issuer. In the Windows’ certificate store this corresponds to “Issued by”. This property can be used to select the signer certificate for signing (see description of Cryptographic Provider in use).

### -cno: Certificate Serial Number

Set the serial number of the certificate. Specify a hex string as displayed by the “Serial number” field in the Microsoft Management Console (MMC), e.g. “49 cf 7d d1 6c a9”. This property can be used to select the signer certificate for signing (see description of Cryptographic Provider in use).

### -cfp: Certificate Fingerprint

Set the hex string representation of the signer certificate’s sha1 fingerprint. All characters outside the ranges 0-9, a-f and A-F are ignored. In the Microsoft Management Console, the “Thumbprint” value can be used without conversion, if the “Thumbprint algorithm” is “sha1”. E.g. “b5 e4 5c 98 5a 7e 05 ff f4 c6 a3 45 13 48 0b c6 9d e4 5d f5”. This property can be used to select the signer certificate for signing (see description of Cryptographic Provider in use).

### -co: Do not Embed Revocation Information

This switch inhibits the embedding of revocation information such as online certificate status response (OCSP - RFC 2560) and certificate revocation lists (CRL - RFC 3280). Revocation information is either an OCSP response or a CRL, which is provided by a validation service at the time of signing and acts as proof that at the time of signing the certificate is valid. This is useful because even when the certificates expires or is revoked at a later time, the signature in the signed document remains valid.

Embedding revocation information is optional but suggested when applying advanced or qualified electronic signatures. If the embedding is enabled then the information of the signer certificate and the issuer certificates other than the root certificate is embedded as well. This implies that both OCSP responses and CRLs can be present in the same message.

The downsides of embedding revocation information are the increase of the file size (normally by around 20k) and that it requires a connection to a validation service, which delays the process of signing (normally by around 2 seconds). For mass signing it is suggested to use the caching mechanism, see chapter "Caching of CRLs, OSCP and TSP Responses".

Embedding revocation information requires an online connection to the CA that issues them. The firewall must be configured accordingly. In case a web proxy is used, it must be ensured the following MIME types are supported when using OCSP (not required for CRL):

```
application/ocsp-request
application/ocsp-response
```

### -cp: Cryptographic Provider

This property specifies the cryptographic provider used to create and verify signatures.

For more information on the different providers available, see the description in the respective subsection of the section Cryptographic Provider.

- When using the Microsoft CryptoAPI Provider, the value of this property with the following syntax:
  `"[ProviderType:]Provider[;PIN]"`
  Examples, "123456" being the pin code:
  - `Provider = "Microsoft Base Cryptographic Provider v1.0;123456"`
  - `Provider = ";123456"`
- When using the PKCS#11 Provider, the value of this property is to be set to a string with the following syntax: `"PathToDll;SlotId;Pin"` Examples:
  - `Provider = "\WINDOWS\system32\siecap11.dll;4;123456"`
- When using any of the service providers, such as the Swisscom All-in signing service, the value of this property is essentially the url of the service endpoint:
  `"http[s]://server.servicedomain.com:8080/url"`

### -cps: Cryptographic session property (string)

String property for configuring cryptographic session. The supported names and values are specific to the cryptographic provider.

### -cpf: Cryptographic session property (file)

File data property for configuring cryptographic session. The supported names and values are specific to the cryptographic provider.

### -csl: Certificate Store Location

For the Microsoft CryptoAPI Provider this defines the location of the certificate store from where the signing certificate should be taken. Supported are:

0 Local Machine

1 Current User (default)

For more information, see the detailed description of the Microsoft CryptoAPI Provider.

### -csn: Certificate Store Name

For the Microsoft CryptoAPI Provider this defines the certificate store from where the signing certificate should be taken. This depends on the OS. The default is "MY". Other supported values are: "CA" or "ROOT".

**Example:** use the certificate store ROOT from the Local Machine account.

```
-cn "..." -csn ROOT -csl 0
```

### -tsu: Time-stamp URL

The URL of the trusted Time-stamp server (TSA) from which a Time-stamp shall be acquired. This setting is only required when applying a Qualified Electronic Signature. Applying a Time-stamp requires an online connection to a time server; the firewall must be configured accordingly. In case a web proxy is used, it must be ensured the following MIME types are supported:

`application/timestamp-query`

`application/timestamp-reply`

### -tsc: Time-stamp Credentials

If a Time-stamp server requires authentication, use this switch to provide the credentials.

**Example:** Credentials commonly have the syntax *username:password.*

```
-cn "..." -tsu http://mytimestamp.com -tsc username:password
```

### -wpu: Web Proxy Server URL

In an organization where a web proxy server is in use, it must be ensured this web proxy server is specified. The URL is something like "http://proxy.example.org" or an IP address. For more information, see the chapter How to Use a Proxy.

### -wpc: Web Proxy Server Credentials

If a web proxy server is used, and it requires authentication, use this switch and the syntax `user:password`.

**Example:** set a web proxy server URL and use authentication.

```
-wpu "http://proxy.example.org" -wpc user:password
```

For more information, see the chapter How to Use a Proxy.

### -nc: Disable cache for CRL and OCSP

Get or set whether to disable the cache for CRL and OCSP responses. Using the cache is safe, since the responses are cached as long as they are valid only. The option affects both signature creation and validation.

See section on caching for more information on the caches.

### -af1: Signature Font Name 1

This is the path to the font name used in upper text, i.e. the text that is set by -at1.

### -af2: Signature Font Name 2

This is the path to the font name used in lower text, i.e. the text that is set by -at2.

### -at1: Signature Text 1

This is the upper text that is added to the signature. If this property is set to blank, the signature name is added to the upper text line of the visual signature.

### -at2: Signature Text 2

This is the lower text that is added to the signature. The text can be multi-lined by using carriage returns. If this property is set to blank, a three-line text is constructed that consists of:

- A statement who applied to signature
- The reason of the signature
- The date

### -abg: Signature Background Image

This is the background image that is added to the signature. The image is centered and scaled down proportionally to fit into the given rectangle. If the path is NULL, or the image does not exist, the appearance's background is a filled rectangle using the colors fill color and stroke color. Note that for the output file to be PDF/A, the image's color space must match the document's output intent.

In order to create a signature with the image only, set the signature texts 1 and 2 to " ".

## 5.5 OCR

In order to make use of OCR, an OCR engine must be installed. The OCR engine is provided as part of a separate product: The 3-Heights™ OCR Enterprise Add-On.

The recommended options (besides `-ocr`, `-ocl` and `-ocp`) are:

- *For scanned documents*: `-oca`
- *For born-digital documents*: `-ocs -oci`

### -le: List OCR Engines

OCR engines are accessed through the corresponding OCR interface DLLs. At present interfacing the following engines is supported:

*Abbyy FineReader OCR Engine*: This engine is accessed by the OCR interface DLL *pdfocrpluginAbbyy.ocr*.

*3-Heights™ OCR Service*: This service is accessed by the OCR interface DLL *pdfocrpluginService.ocr*. The service accesses the Abbyy FineReader 8.1 OCR Engine.

*Tesseract*: This engine is accessed by the OCR interface DLL *pdfocrpluginTesseract.ocr*.

The OCR interface DLLs are provided by the 3-Heights™ PDF to PDF/A Converter Shell. The OCR engine is provided as a separate product, such as 3-Heights™ OCR Enterprise Add-On. Here is an example of listing available OCR engines:

```
-le
List of available OCR engines:
- abbyy
- abbyy10
- service
- tesseract
End of list.
```

In order to make use of the OCR engine, the OCR interface DLL and the OCR engine must be installed. The switch -le lists all available OCR interface DLLs. It does not verify the corresponding OCR engine is installed and can be initialized. The OCR engine is actually accessed when using the switch -ocr.

### -ocr: Load OCR Engine

If a PDF document has to be made fully text searchable even if the text is part of a raster image then the images which are contained in the PDF document must be run through an OCR engine. With this switch the user can select an OCR engine, e.g. "Abbyy", and instruct the tool to embed the recognized text as a hidden layer on top of the image. If the add-in is not found or the engine cannot be initialized (because it is not installed or the license key is not valid) then an error message is issued.

The name of the OCR engine can be retrieved using the switch -le. If the switch -ocr is not used, no OCR is applied. The following switch sets the OCR engine to Abbyy:

```
pdf2pdf -ocr abbyy input.pdf output.pdf
```

See also documentation for the 3-Heights™ OCR Add-On.

### -ocl: Set OCR Language

In order to optimize the performance of the OCR engine, it can be given hints what languages are used. The default language of the Abbyy FineReader 8.1 OCR Engine is English. This switch can only be used if the switch -ocr is set. This setting depends on the OCR engine, e.g. it is different for Abbyy and Tesseract. The following switch set the languages to English and German:

```
pdf2pdf -ocr abbyy -ocl "English, German" input.pdf output.pdf
```

See also documentation for the 3-Heights™ OCR Add-On.

### -ocp: Set OCR Parameters

Using this switch OCR engine specific parameters (key/value pairs) can be set to optimize the performance. The following switch enables the Balanced Mode to improve the speed and do not detect whether text is bold or not.

```
pdf2pdf -ocr abbyy -ocp "BalancedMode=TRUE, DetectBold=FALSE" input.pdf output.pdf
```

See also documentation for the 3-Heights™ OCR Add-On.

### -ocs: Do Not Re-embed De-skewed Image

The OCR engine de-skews and de-noises the input image before recognizing the characters. This option controls, whether the 3-Heights™ PDF to PDF/A Converter Shell should use the de-skewed image or keep the original image.

- *With option -ocs:* Embed the original image (also see option -oci). This setting is recommended for born-digital documents.
- *Without option -ocs:* Embed the de-skewed and de-noised image from the OCR engine. This might change the appearance of the page. This setting is recommended for scanned documents.

### -oci: Do not deskew image

Do not de-skew original image (with -ocs only). This option specifies whether the image and text are de-skewed according to the recognized skew angle.

- *With option -oci:* Do not change skew of images (i.e. do not change appearance of the page). This setting is recommended for born-digital documents.
- *Without option -oci:* Rotate image, such that lines of text are made horizontal. This might change the appearance of the page. This setting is recommended for scanned documents.

### -ocd: Resolution for OCR Recognition

Resample images to target resolution before they are sent to the OCR engine. If no value is set, images are re-sampled to 300 dpi for OCR, which is the preferred resolution for most OCR engines.

### -oct: Threshold Resolution for OCR

Only images with a higher resolution than the threshold are re-sampled before OCR. The default is 400 dpi. If set to -1: no re-sampling is applied.

**Example:** Resample all images with a resolution of more than 300 dpi to 300 dpi:

```
-ocd 300 -oct 1
```

**Example:** Resample all images with a resolution of 400 dpi or more to 300 dpi (default):

```
-ocd 300 -oct 400
```

**Example:** Do not resample:

```
-oct -1
```

Compatibility Note: Initially this switch was called -ocD and then renamed to -oct to avoid confusions with the switch -ocd.

### -ocb: Convert Images to Bitonal before OCR Recognition

Specify whether the images should be converted to bi-tonal (black and white) before OCR recognition.

Enabling this feature can improve the memory consumption of the OCR process. It is suggest to use this feature with ABBYY 8 or Tesseract.

Enabling this feature automatically re-embeds the original images in the output document; the setting of -ocs is ignored.

### -ocm OCR mode

Specify behavior of converter for files with existing OCR text. Available OCR modes are the following:

1. Only perform OCR for images without existing OCR text (default).
2. If OCR engine is active, remove old OCR text and perform OCR for all images. Hence, existing OCR text is not removed, if OCR engine is not active.
3. Always remove old OCR text and, if OCR engine is active, perform OCR for all images. This can be used to strip existing, without adding new OCR text.
4. Only perform OCR, if input file contains no text.

**Example:** Set OCR Mode 2

```
-ocm 2
```

### -oca: Rotate the image according to the detected angle

The OCR engine may detect that an image needs to be rotated in order to have the text in an up-right position. If this is the case and this switch is used then the original image is replaced by the rotated image.

### -ocbc: Embed barcodes

Embed the recognized barcodes in the XMP metadata.

### -ocx: Export recognized ocr text to file

Export the retrieved OCR text to a file. This function can only be used in combination with an OCR engine (see option -ocr). When an OCR engine is set, the OCR text is always embedded in the resulting PDF document. If this method is used, it is in addition also extracted to a file. The output format is a table, where rows are separated by a new line and columns are separated by a tabulator. The table contains the following columns:

| | |
|---|---|
| `Page` | Page number |
| `Image` | PDF object number which contains the image |
| `FontSize` | Font size in points |
| `FontName` | Font name, for any barcode font the name is “Barcode”. This value is only set if the font name is returned by the OCR engine. |
| `FontFamily` | 1: Serif<br>2: SansSerif<br>3: Monospaced<br>This value is only set if provided by the OCR engine. |
| `FontStyles` | 1: Barcode<br>2: Bold<br>4: Italic<br>8: Underline<br>16: Strikeout<br>This value is only set if provided by the OCR engine. Example: 6 = 2 + 4 = Bold + Italic |
| `Baseline` | Baseline of the text |
| `Left, Top, Right, Bottom` | Bounding box of the text in PDF coordinates |
| `String` | Recognized text |

**Example:** Write extracted text to the text file “text.txt”.

```
pdf2pdf -ocr abbyy -ocx text.txt input.tif output.pdf
```

## 5.6 Return Codes

There are different return codes supported. All but return code 0 indicate an error or problem.

0 Success

Occurs when:
- Analysis only and no errors
- Conversion without errors
- Conversion and post-analysis without errors

1 Cannot open input file

2 Cannot create output file

3 Parameter error

Examples:
- File does not exist
- Invalid XMP file
- File is not a valid ICC color profile
- ICC version is not compliant
- Other parameter error

4 Cannot convert input file due to conformance problems

5 Output file has conversion errors

A conversion error means the input document contains an element that does not exist in PDF/A, i.e. can only be converted in a way that the result may have visual or other differences. However the resulting document is PDF/A compliant. This return code depends on the setting -cem.
- Missing output intent of device color space
- Optional content removed
- FFilter or FDecodeParms removed
- Prohibited annotation type converted to text annotation
- Prohibited action removed
- Embedded files removed
- Transparency removed
- Character from show string removed because glyph missing in font

Some of these conversion errors, such as transparency or optional content may be resolved by creating PDF/A-2 or PDF/A-3 instead of PDF/A-1.

6 Output file is not conformant (post analysis)

Occurs when:
- Post analysis failed

Examples:
- Missing output intent of device color space
- Font not embedded
- Annotation without appearance stream

7 There was a problem during the linearization of the file

The following return codes became deprecated with version 2.1.25.0. Instead 5 is returned, depending on the setting to mask conversion errors.

8 Input file is not conformant (pre analysis)

Occurs when:
- Analysis only with errors
- Conversion with failed pre analysis but successfully post analysis

9 Parts of the XMP metadata could not be repaired and had to be removed.

11 Input document is not PDF/A compliant and contains a digital signature and the switch -ca is set

12 A signature creation error occurred. Possible reasons are:
- Cannot create a session (or CSP)
- The certificate store is not available
- The certificate cannot be found
- The private key is not available
- Incorrect signature length

13 Cannot get response from OCSP or TSP

If the input file cannot be read, the following error is returned (return code 1):

```
0x80410101 - E - The file could not be opened.
Couldnt open input file input.pdf.
```

If the output file cannot be created, the following error is returned (return code 2):

```
0x80410102 - E - The file could not be created.
Couldnt create output file output.pdf.
```

# 6 Log File

All steps in the diagram from chapter "Process Description" can write to the log file. There are three types of messages in the log file: Warnings/Information, Errors and Reports.

## Warnings and Information

describe the current process step. They do not inhibit the conversion. Prefix: -

### Example:

```
- Opening file input.pdf
- Setting font directory to C:\WINNT\Fonts
- Analyzing input.pdf
- Conformance level A has been downgraded to level B
```

## Errors

inhibit a successful conversion. Prefix: *

### Example:

```
* Cannot open file input.pdf.
Input file 001.pdf isn't convertible.
```

### Reports

are only created if the corresponding option (Details or Summary) is selected. Prefix: none

#### Example: Details

```
0, 0x80410604, "The key Metadata is required but missing.", 1
2, 0x00418704, "The font Verdana must be embedded.", 1
2, 0x83410614, "The document contains device-specific color spaces (Annotation C or IC).", 1
```

#### Example: Summary

```
The document contains fonts without embedded font programs or encoding information (CMAPs).
The document's meta data is either missing or inconsistent or corrupt.
The document doesn't provide appropriate logical structure information.
```

# 7 Licensing and Copyright

The 3-Heights™ PDF to PDF/A Converter Shell is copyrighted. This user's manual is also copyright protected; it may be copied and given away provided that it remains unchanged including the copyright notice.

# 8 Contact

PDF Tools AG

Kasernenstrasse 1

8184 Bachenbülach

Switzerland

http://www.pdf-tools.com